VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

METODY DITHERINGU OBRAZU

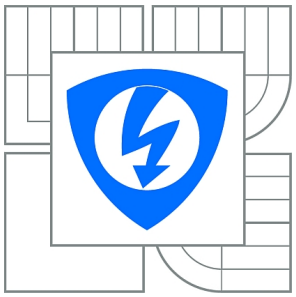
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. LUKÁŠ PELC
AUTHOR

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

METODY DITHERINGU OBRAZU

METHODS OF IMAGE DITHERING

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

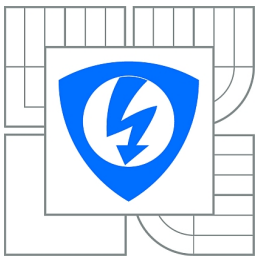
AUTOR PRÁCE — Bc. LUKÁŠ PELC

AUTHOR

VEDOUCÍ PRÁCE — Mgr. PAVEL RAJMIC, Ph.D.

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Lukáš Pelc	***ID:***	109708
Ročník:	2	***Akademický rok:***	2013/2014

NÁZEV TÉMATU:

Metody ditheringu obrazu

POKYNY PRO VYPRACOVÁNÍ:

Nastudujte teoreticky běžné i moderní metody pro dithering obrazu. Naprogramujte vybrané metody v Matlabu nebo jako JAVA applety. Porovnejte výsledky subjektivním testem i objektivními metodami. Porovnejte rovněž výpočetní náročnost.

DOPORUČENÁ LITERATURA:

[1] Jiří Žára, Bedřich Beneš, Jiří Sochor, Petr Felkel, Moderní počítačová grafika, druhé vydání, Computer Press, 2005, ISBN 80-251-0454-0

[2] R. C. Gonzales, R. E. Woods, Digital Image Processing, Third Edition, Prentice Hall, 2008

[3] Lau, D.L., Arce, G.R., Modern Digital Halftoning. CRC Press, druhé vydání, 2008, ISBN 1420047531

Termín zadání:	10.2.2014	***Termín odevzdání:***	30.5.2014

Vedoucí práce: Mgr. Pavel Rajmic, Ph.D.
Konzultanti diplomové práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Diplomová práce pojednává o metodách ditheringu obrazu. Základem je vysvětlení teorie digitálního obrazu, barevných modelů, palety a hloubky barev. Následuje rozebrání základních metod omezování barevného prostoru jakou jsou metody prahování, náhodného a maticového rozptylu. Rozebrány jsou i pokročilé metody ditheringu s distribucí chyby, v čele s nejznámější metodou Floyd-Steinberg. Zahrnuto je porovnání jednotlivých metod včetně subjektivního srovnání pomocí dotazníku. Programovou část tvoří JAVA applet, který ukazuje možnosti generování obrázků pomocí jednotlivých metod ditheringu.

KLÍČOVÁ SLOVA

Omezení barevného prostoru, barevná hloubka, barevný model, obraz, prahování, rozptyl, distribuce chyby, java.

ABSTRACT

Master's thesis discusses methods for dithering image. The basis is the explanation of the theory of digital images, color models, color depth and color range. Followed by the dismantling of the basic dithering methods which are a thresholding method, a random and matrix diffusion. Discussed are advanced methods of dithering with error distribution, bee with best known method Floyd-Steinberg. Included is a comparison of different methods including subjective comparison using a questionnaire. Program part is JAVA applet that shows the possibility of generating images using various dithering methods.

KEYWORDS

Dithering, color depth, color model image, tresholding, diffusion, error distribution, java.

PELC, Lukáš *Metody ditheringu obrazu*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2012. 74 s. Vedoucí práce byl Mgr. Pavel Rajmic, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma Metody ditheringu obrazu jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Mgr. Pavlu Rajmicovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

(podpis autora)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .
(podpis autora)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

V diplomové práci se věnuji metodám ditheringu obrazu. V úvodní části rozebírám základní pojmy, které je potřeba si pro pochopení ditheringu objasnit. Mezi tyto základní pojmy patří digitální obraz a jeho formy, srovnání jednotlivých barevných modelů. Objasnění pojmů jako je barevná hloubka, barevná paleta, či kvantování barev. V závěru kapitoly je nastíněno, co je vlastně dithering obrazu a jaké jsou možnosti jeho využití.

Dále jsou v práci rozebrány jednotlivé metody ditheringu. Mezi ty základní patří prahování, náhodný a maticový rozptyl. Další metody již můžeme zařadit mezi tzv. moderní metody ditheringu. Jsou to metody s distribucí zaokrouhlovací chyby, z nichž nejznámější je metoda Floyd-Steinberg. Následuje objasnění funkčnosti a srovnání dalších metod s distribucí chyby.

V třetí části diplomové práce šlo o naprogramování java-appletu. Java-applet by měl obsahovat vybrané metody ditheringu obrazu s nastavením jejich funkčnosti. Možnost generovat jednotlivými metodami výsledné obrázky s následným uložením do souboru.

Závěrečná část práce se zaměřuje na porovnání jednotlivých metod ditheringu. Především je to subjektivní vnímání obrázků generovaných jednotlivými metodami. Přičemž do srovnání se použijí obrázky vytvořené pomocí java-appletu. Srovnání bude provedeno pomocí dotazníku. Zahrnut by měl být i vlastní názor na výsledky jednotlivých metod.

1 TEORETICKÝ ÚVOD

V této kapitole se zaměřím na vysvětlení základních pojmů týkajících se digitálního obrazu. Především jak se obraz tvoří a z čeho je složen. Objasním zde pojmy jako je barevný prostor, barevná hloubka a paleta. Na závěr kapitoly vysvětlím, proč vlastně potřebujeme omezovat barevný prostor, nebo-li k jakému účelu se tato možnost změny obrazu využívá.

1.1 Digitální obraz

Digitální obraz je binární reprezentace dvojrozměrného obrazu v počítači. Digitální obraz v počítači může být buď rastrový a nebo vektorový. Oba typy jsou vysvětleny níže. V práci se budu věnovat obrazu rastrovému, tedy tvořenému čtvercovými body – pixely.

1.1.1 Rastrový obraz

Rastrový obraz je tvořen pomocí obrazových bodů – pixelů. Tyto body jsou uspořádány do rastrové mřížky a reprezentují hodnotu světelné intenzity původního obrazu. Každý bod má v mřížce přesně určenou polohu. Díky tomu jsme schopni jednotlivé světelné intenzity vyjádřit jako diskrétní matici, se kterou můžeme dále pracovat.

Rastrová grafika se také označuje jako bitmapa – bitová mapa. Bitmapa by měla interpretovat obraz složený z pixelů o dvou stavech (1 bit), tedy pixel vykreslený nebo nevykreslený (monochromatický obraz). Pokud má obraz větší bitovou hloubku barev, měl by být označován jako pixmapa.

Mezi hlavní výhody rastrové grafiky patří snadnost jejího pořízení. Sloužit k tomu může digitální fotoaparát nebo skener. Nevýhodou je vysoká datová náročnost v profesionální grafice, kdy při vysokém rozlišení a vysoké barevné hloubce dosahuje velikost obrazů i desítek megabytů. Další nevýhodou je, že při změně velikosti obrazu dojde ke zhoršení obrazové kvality. Nejvíce je to patrné při zvětšování, kdy je viditelný rastr obrazu. [1]

1.1.2 Vektorový obraz

Vektorový obraz je geometrickou interpretací křivek – vektorů. Jedná se o Bézierovy křivky. Křivky jsou popsány pomocí čtyř bodů, dvou krajních bodů (kotevní body) a dvou bodů určující tvar křivky (kontrolní body). Pomocí těchto křivek lze interpretovat jakékoliv geometrické tvary, jako jsou body, přímky, mnohoúhelníky a další.

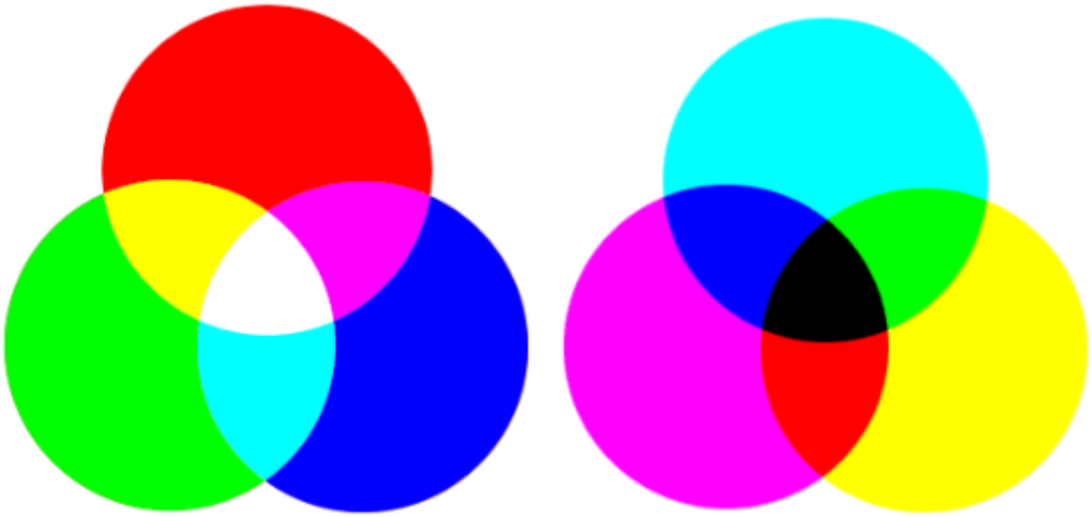
Mezi výhody vektorové grafiky patří libovolná změna velikosti obrazu bez ztráty kvality. Dále možnost pracovat s každým objektem zvlášť a také mnohem menší datová náročnost, než u grafiky rastrové. Nevýhodou je složitost pořízení obrazu, grafické objekty se musí vytvořit. Další nevýhodou je náročnost na zpracování v počítači pokud se jedná o složitější grafické objekty. [1]

1.2 Barevný model

Barevný model popisuje základní barvy z nichž je míchána výsledná barva jednotlivých bodů obrazu. Barva je subjektivní vnímání viditelného spektra elektromagnetického záření, neboli světla. Světlo je spojitý signál složený z mnoha frekvencí, jež je potřeba diskreditovat a modelově popsat.

Světlo je elektromagnetické vlnění s různou vlnovou délkou. Lidské oko není schopno zachytit všechny tyto vlnové délky, ale pouze oblast viditelného světla o délce přibližně 400–750 nm. V této vlnové délce je lidské oko schopno rozlišit velké množství barev. Není ovšem pravda, že každá barva má jinou vlnovou délku. Lidské oko totiž rozlišuje kromě odstínu barvy také její jas a sytost. Různou vlnovou délku tedy mají barvy s jiným odstínem

Existují dva základní způsoby míchaní barev – aditivní (sčítací) a subtraktivní (odečítací) barevný model, viz. obrázek 1.1. Při aditivním barevném modelu se od černé barvy přičítají jednotlivé barevné složky. Při maximální intenzitě všech barevných složek vzniká barva bílá. U subtraktivního modelu je to opačně, při maximální intenzitě tedy vzniká teoreticky černá. [12]



(a) Aditivní (b) Subtraktivní

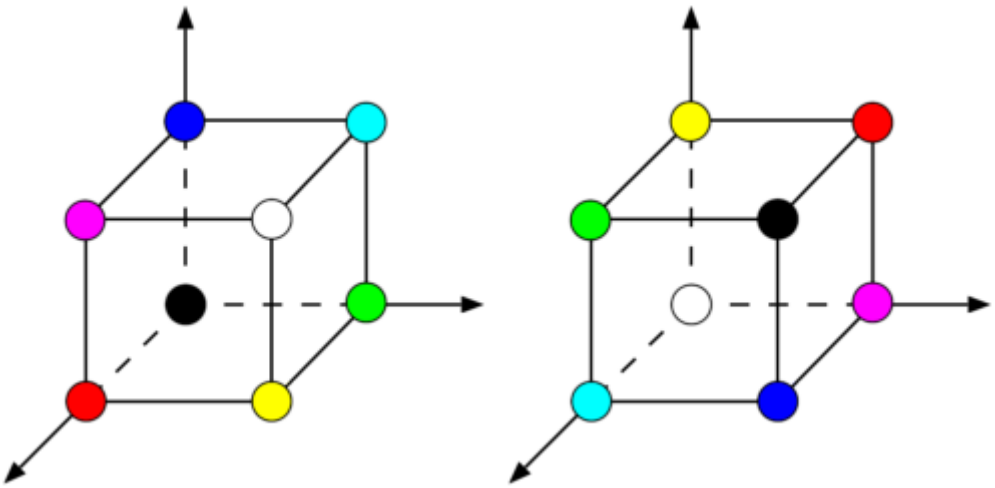
Obr. 1.1: Míchání barev

1.2.1 RGB

Barevný model RGB (Red, Green, Blue) je aditivní barevný model. Výsledná barva je tedy součet jednotlivých barevných složek. Vytváří se světlo větší intenzity.

Tento barevný model je tvořen třemi barevnými kanály – červeným (Red), zeleným (Green) a modrým (Blue). Vychází z vnímání barev lidským okem, které obsahuje tři druhy čípků lišících se barevným pigmentem. Ty odpovídají vlnovým délkám červeného, zeleného a modrého světla. Základem toho modelu je kostka v jejímž počátku (0,0,0) je černá barva. Protilehlý vrchol má barvu bílou (1,1,1), kdy jsou všechny barvy v nejvyšších intenzitách. Ostatní barvy se vytvářejí mícháním různých intenzit barevných složek. Barevný model je vidět na obrázku 1.2.

Tento barevný model se používá u digitálních zobrazovacích zařízení, např. monitorů. Jednotlivé pixely monitoru jsou složeny ze tří RGB diod. Zhasnutý monitor má černou barvu, rozsvěcením jednotlivých diod v obrazovém bodě docílíme požadované barvy.[12]



(a) RGB (b) CMY

Obr. 1.2: Barevné modely RGB a CMY

RGBα

Barevný model RGBα vychází z barevného modelu RGB. Jedná se tedy také o aditivní barevný model, jenž je doplněný o α-kanál, který obsahuje informaci o průhlednosti daného obrazového bodu. Obrazový bod o intenzitě 0% má maximální průhlednost – je tedy neviditelný. Naopak intenzita α-kanálu 100% značí maximální neprůhlednost. Tento barevný model je vhodný obraz složený z více vrstev. [12]

1.2.2 CMY

Barevný model CMY (Cyan, Magenta, Yellow) je subtraktivní model míchání barev. Jednotlivé barevné složky se od původní barvy odečítají a omezují barevné spektrum. Vyšší intenzitou jednotlivých barev se ubírá světlost barvy původní, tedy bílé. Při maximální intenzitě všech tří složek vzniká teoreticky černá.

Barevný model CMY je doplňkový model k modelu RGB. Výsledná barva CMY se dá vypočítat odečítáním RBG barev od jednotkové krychle, viz obrázek 1.2.

Využití tohoto barevného modelu najdeme především u tiskáren. Papír vkládaný to tiskáren je bílý a postupným mícháním barev CMY vytváříme barvy další, přičemž pokud použijeme všechny tři barvy o maximální intenzitě dostaneme barvu černou. [12]

CMYK

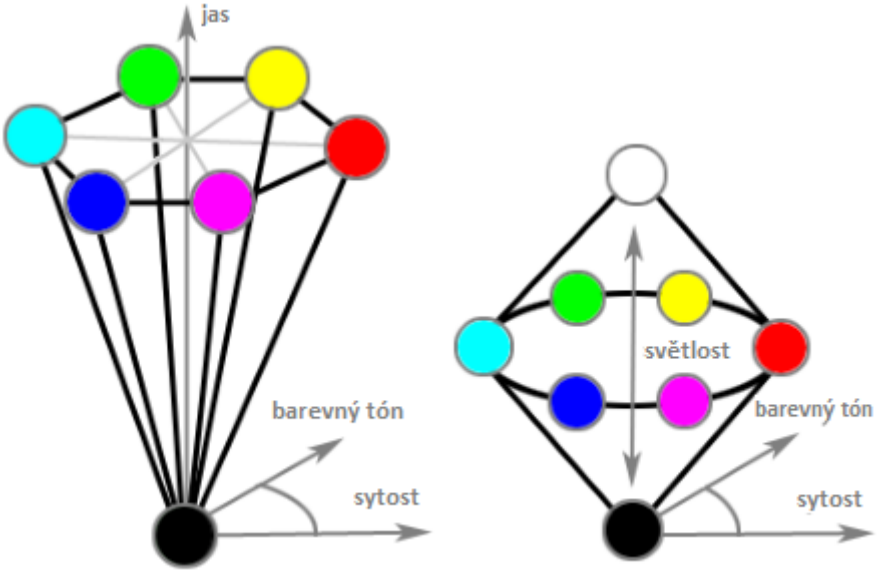
Barevný model CMYK vychází z modelu CMY, je ale doplněný o černou barvu. Původní CMY model měl nedostatek v krytí jednotlivých barevných složek. U tiskáren nastávaly dva problémy. První z problémů bylo přesné krytí všech tří složek CMY nad sebou a druhý problém, že i když se to podařilo, výsledná barva nebyla černá, ale spíše tmavě šedá. Proto se začala používat ještě černá barva, která umožňuje i ztmavení odstínů. Z ekonomického hlediska je lepší mít černou kazetu zvlášť, než míchat černou barvu ze tří kazet, spotřebuje se třikrát méně inkoustu. V dnešní době už existují i více kazetové tiskárny, které mají přidanou třeba světle oranžovou kazetu pro lepší míchání pleťových odstínů na fotografiích. [12]

1.2.3 HSV

Model HSV (Hue, Saturation, Value), známý také jako model HSB(Hue, Saturation, Brightness). Tento model byl vytvořen tak, aby odpovídal míchání barev jak jej vnímá lidské oko. Obsahuje tři složky: odstín, sytost a jas.

Tento barevný model představuje obrácený kužel. Odstín barvy se vybírá na kruhové podstavě. Sytost barvy, je vlastně množství šedi v poměru k odstínu barvy. Je to hodnota v procentech od 0% (šedá) až po 100% (plně sytá barva). V kruhu je největší systost na jeho okrajích. Poslední hodnotou modelu je jas. Nejnižší hodnotu jasu představuje vrchol kuželu a nejvyšší hodnota se nachází v kruhové podstavě. Barevný model je vidět na obrázku 1.3.

Největší výhodou tohoto barevného modelu je, že se při manipulaci s barvou volí nejdříve barevný odstín a potom sytost a jas. To znamená, že při manipulaci se sytostí a jasem nedochází ke změně odstínu barvy. U modelů RGB a CMY toho dosáhneme velmi těžko. [12]



(a) HSB (b) HSL

Obr. 1.3: Barevné modely RGB a CMY

1.2.4 HSL

Barevný model HSL (Hue, Saturation, Lightness) je velmi podobný předchozímu modelu HSV. Tento model odpovídá nejvíce skutečnosti, protože je jas nahrazen světlostí. Schopnost rozlišovat barevné odstíny totiž klesá/stoupá se ztmavením/ zesvětlením základní barvy. K tomu dochází zvyšování a snižováním světlosti barvy. Tím odpovídá tento model nejvíce možnostem vnímání barev lidským okem. [12] V porovnání na obrázku 1.3.

1.3 Barevná hloubka

Barevná hloubka určuje množství barevné informace, které nese jeden obrazový bod obrazu. Je to počet bitů určujících barvu jednoho pixelu v obraze, známá jako počet bitů na pixel (bpp). Vyšší barevná hloubka udává větší rozsah barev a zároveň zvyšuje množství datových informací v obraze. Základní používané barevné hloubky jsou v tabulce 1.1

Tab. 1.1: Barevné hloubky využívané v počítačové grafice

Barevná hloubka [bity]	Počet barev	Název palety
1	$2^1 = 2$	Mono color
2	$2^2 = 4$	CGA
4	$2^4 = 16$	EGA
8	$2^8 = 256$	VGA
15	$2^{15} = 32768$	Low color
16	$2^{16} = 65536$	High color
24	$2^{24} = 16777216$	True color
32	$2^{32} = 4294967296$	Super true color
48	$2^{48} = 281474976710656$	Deep color

1.4 Barevná paleta

Barevnou paletou rozumíme množství barevných odstínů, jenž mohou jednotlivé pixely nabývat. Palety barev souvisejí s použitým barevným modelem, v něm jsou uložené hodnoty reprezentovány. Každý pixel obrazu potom uchovává index odkazující na barvu v barevné paletě.

V případě RGB palety jsou to tři kanály – červený, zelený a modrý. Jednotlivé pixely obrazu mají daný rozsah hodnot, který vypovídá o intenzitě barevného kanálu. Jako nejjednodušší příklad lze uvést 1bitový rozsah. Pro každý kanál se uchovávají dvě hodnoty 0, nebo 1. Složením kanálů RGB vzniká 3bitová paleta, ze které je možno mícháním barev dosáhnout 8 odstínů. V tomto případě se jedná o uniformní rozložení. Pomocí níže zmiňovaných metod ditheringu lze potom navodit i vjem více-barevné palety. Dále zmiňované barevné palety budou rozebrány v prostoru RGB. [12, 5]

Paleta 3-3-2

Neuniformní rozložení prostoru pro všechny kanály využívá právě paleta 3-3-2. Toto rozložení přiděluje menší rozsah pro modrý kanál, na který je lidské oko nejméně citlivé. Rozdělení 3-3-2 je zvoleno tak, aby se do jedno pixelu vešlo 8 bitů. Výsledkem je třetinová paměťová náročnost oproti 24bitové paletě. [13]

N-prvková paleta

Redukce na N-prvkovou paletu se provádí následovně. Původní obraz v barevné hloubce například 24bitů nese pro každý kanál informaci v 8 bitech (tedy 256 odstínů). Pokud tento obraz budeme chtít redukovat na nějakou paletu s menším

počtem barevných odstínů, musíme přistoupit na redukci barev. Použijeme opět nejjednodušší příklad, zůstaneme u 3bitové palety. Pro každý kanál RGB musíme z 256 hodnot vytvořit dvě. Redukce je tedy z 8 bitů na 1 bit. K tomu je možné použít prahování, kdy dojde k rozhodnutí o barvě daného pixelu na základě toho, jestli je je pod úrovní, nebo nad úrovní prahu. Více v kapitole 2.1. Při více-barevné paletě se pouze rozhoduje mezí více hodnotami prahů.

Tímto postupem se kvantují jednotlivé barevné složky a až se tak stane, výsledný obraz se získá zpětným složením RGB kanálů. Výsledný obraz má tak redukovanou paletu barev. Opět se dá použít metod ditheringu pro simulaci více barevných odstínů.

Dithering se u těchto obrazů aplikuje na každý barevný kanál zvlášť. Při distribuci chyby se také vypočítává a přenáší hodnota chyby pro jednotlivé složky.

Indexovaná paleta

Indexovaná paleta barev přináší výhodu oproti předchozím paletám. Zaměřuje se na redukci paměťových nároků na uložení obrazu.

Principem této palety je vygenerování odstínů základní RGB palety. Potom se pixel po pixelu obraz namapuje na základní paletu tak, že každý pixel obsahuje index do základní palety. Tím se dosáhne značné úspory paměti, protože každý pixel nese pouze index do základní palety. Nevýhodu je stále omezený počet barev palety a opět množství v obraze nepoužitých barev. Tento problém řeší až paleta adaptivní.

Adaptivní paleta

Adaptivní paleta je speciálním případem palety pro konkrétní obraz. Je uložená spolu s obrazem v jednom souboru. Základní vlastností této palety je, že redukuje nadbytečnost barev obsažených v paletě. Adaptivní paleta je stále omezená paleta barev, ale negenerují se v ní všechny barvy, ale především ty, které jsou v obraze uloženy. Předchozí palety totiž obsahují i barvy, které se v obraze nevyskytují. Pokud se nebudeme bavit o manuální přidaní určité barvy do palety, tak dochází k analýze vstupního obrazu, přičemž jsou následně vybrány nejvhodnější barvy pro sestavení palety. Daný prostor palety je tedy zaplněn odstíny barev, které jsou ve výsledném obraze užity. Pro zlepšení kvality výsledného obrazu se potom ještě používá ditheringu, pro simulaci dalších barevných odstínů, které už v paletě obsaženy nejsou. [13]

Převod na stupně šedi

V této práci se budu zabývat především obrazem ve stupních šedi. Obraz se do stupňů šedi převádí následujícím vztahem, podle kterého se počítá výsledná intenzita barev:

$$Y = 0,2989R + 05866G + 0,1144B. \tag{1.1}$$

Tento vztah je dám tak, aby odpovídal vnímání barev lidským okem. V lidském oku se nacházejí světlocivné buňky – tyčinky a čípky. O vjem barvy se starají čípky – červené, zelené a modré. Nejvíce je zelených čípků a nejméně modrých, čemuž odpovídají koeficienty daného vztahu. Nejčastější barevnou hloubkou je 8bitová, která zajišťuje pro oko dostatečných 256 odstínů šedé barvy. [12] Výsledný obraz bude nejčastěji reprezentován monochromaticky.

1.5 Kvantování barev

Kvantování barev je ztrátová barevná komprese. Snižuje se při ní počet barevných odstínů v obraze. Vše za účelem zachování co nejlepší vizuální podoby obrazu původního. Je to ztrátový a nezvratný proces. Nejčastěji se s touto metodou setkáme u rozptylových metod ditheringu obrazu, které popisuji níže.

Při kvantování barev vzniká signál, který má nespojitý průběh, mění se skokem a má omezený počet barevných odstínů. Obraz, neboli obrazová funkce je rozdělena na intervaly, každému intervalu je potom přidělena určitá hodnota.

Existují dva druhy kvantování – unimorfní a neunimorfní. Při unimorfním kvantování je dodržena stále stejná délka intervalu, naopak u neunimorfního kvantování je délka intervalu různá. Výhodou neunimorfního kvantování je možnost přizpůsobení se nerovnoměrnosti rozložení hodnot.

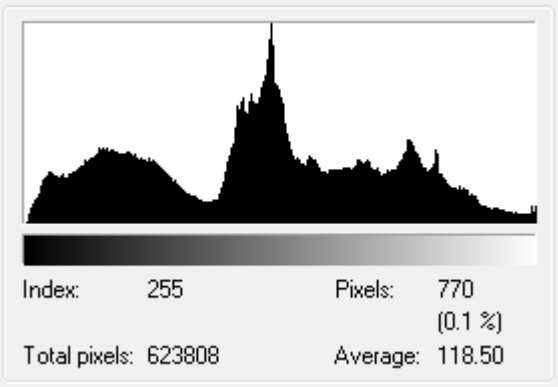
Při kvantování dochází ke kvantizační chybě – tedy ztrátě informace způsobené skokovou změnou barevného přechodu. To se projevuje v plochách s malou změnou gradientu jako náhlý přechod barvy. Vše je založeno na nedokonalosti lidského oka, které není od určité vzdálenosti tyto změny zaznamenat.[11]

1.6 Histogram

Jeden ze způsobů interpretace rozložení barev v obraze je histogram. Histogram je statistický útvar, který zobrazuje pravděpodobnost rozložení určité veličiny v grafu.

Jedním z histogramů používajících se v digitální fotografii je histogram jasový. Z toho histogramu snadno poznáme přeexponovaná a podexponovaná místa v pořízené digitální fotografii. Na ose x jsou znázorněny hodnoty jasu a na ose y jejich relativní četnost, což ukazuje obrázek 1.4.

V případě histogramu barevných složek jsou na ose x hodnoty barevného kanálu od 0 do maximální hodnoty dané velikostí barevné palety (např. pro 8bitovou paletu je to 256 hodnot). Na osy jsou vyneseny hodnotou stejný relativní počet pixelů. [1]



Obr. 1.4: Ukázka histogramu

1.7 Omezení barevného prostoru

Omezení barevného prostoru, neboli dithering je technika používaná při pracování digitálního obrazu. Jedná se o redukci počtu barev v obraze, které má za cíl maximální věrnost původnímu obrazu při minimální paměťové náročnosti. Při ditheringu obrazu se nemění jeho původní rozlišení, pouze se hodnoty každého pixelu nahrazují jinými hodnotami podle zvolené metody. V případě digitálního obrazu se jedná o napodobení barvy pomocí kombinace barev z barevné palety menšího rozsahu.

Existuje několik metod ditheringu, všechny nějakým způsobem využívají nedokonalosti vnímání lidského oka. Základním nedostatkem lidského oka, je že nerozlišuje jednotlivé body obrazu, ale ze shluků bodů vytváří výsledný vjem (např. barvu). Využívá se toho hned v několika případech. Například vímání šedé barvy. Pro lidské oko lze vytvořit vjem šedé barvy pouze vhodnou kombinací bílých a černých bodů. Tato technika se používá u černobílých laserových tiskáren. Nebo další příklad – barva jednoho pixelu počítačového monitoru je vytvářena pouze ze tří RGB složek. V monitorech potom ještě různé techniky pro navození pocitu větší barevné hloubky obrazu. [1]

Polotónování

Polotónování (anglicky Halftoning) je metoda, která nepatří pod dithering, nicméně ji krátce zmíním. Tato metoda je používaná v tisku, kdy se pomocí různě velkých černých skvrn simuluje odstín původního pixelu. Každý pixel obrazu je nahrazen čtvercovou maticí o několika bodech, např. o rozměrech 4×4 pixely. Barva jednotlivých bodů odpovídá prahování vstupního bodu. Větší hodnoty prahu jsou na okrajích matice, tedy čím tmavší pixel tím více vykreslených bodů v okrajích matice.

Největší výhodou této metody je, že jsme schopni tiskovou hlavou s pouze černou barvou vytvořit velký rozsah intenzit daný velikostí matice. Tedy čím větší matice tím více úrovní šedé jsme schopni nasimulovat.

Důvodem proč nezařazuji tuto metodu mezi dithering je, že v důsledku zpracování obrazu dochází ke zvětšení rozlišení. U ostatních metod zůstává rozlišení obrazu vždy stejné jako v obraze původním. Nicméně toto není pro dnešní tiskárny žádný problém, protože mají mnohem větší hodnotu DPI než běžné LCD displeje. [2]

2 METODY DITHERINGU OBRAZU

Tato část diplomové práce pojednává o metodách ditheringu obrazu. Jedná se o metody běžné i moderní, např. metody distribuce zaokrouhlovací chyby. Metody budou demonstrovány na obrázcích ve stupních šedi, přičemž výsledná obraz bude většinou v 1bitové paletě barev – tedy monochromatický obraz. Toto řešení se jeví jako nejlepší pro demonstraci výsledných obrazů. Použití vyšší barevné hloubky má za následek problémy jak se zobrazením, tak s tiskem.

2.1 Prahování

Prahování je nejzákladnější metoda pro omezení barevného prostoru na jednobitovou paletu barev.Je to metoda velmi rozšířená díky své jednoduchosti.

Principem metody je rozdělení palety barev pomocí prahu. Jedná se vlastně o nastavení prahové hodnoty v histogramu obrazu. Hodnotách pixelů nacházejících se vlevo od prahové hodnoty se přiřadí černá barva, naopak hodnotám nacházejícím se vpravo se přiřadí bílá barva.

Při prahování dochází k transformaci vstupního obrazu $f(i,j)$ a na výstupní $g(i,j)$ pomocí rovnice:

$$g(i,j) = \begin{cases} 1 \text{ pro } f(i,j) & > T \\ 0 \text{ pro } f(i,j) & \leq T \end{cases} . \tag{2.1}$$

Hodnota $g(i,j) = 1$ určuje černou barva, hodnoty $g(i,j) = 0$ je pro bílou barvu. Rozhodování mezi černou a bílou barvou je na základě zvolené prahové hodnoty T.

Nejlepší výsledky tato metoda podává ve vysokých frekvencích – zvýrazňuje hrany objektů. Méně přesvědčivé výsledky tato metoda podává v nízkých frekvencích, jako jsou jemné přechody v obraze. Využití tedy najdeme například v programech pro rozpoznávání písmen, ve snímačích otisků prstů a rozpoznávání objektů v obraze. Prahování také umožňuje odstranění nežádoucích vlivů v obraze, třeba vrhaný stín.

Základní problematikou prahování je je nalezení optimální prahové hodnoty. Metody jak docílit nalezení prahu jsou manuální, automatická a adaptivní.

2.1.1 Manuální práh

Nejjednodušší způsob stanovení prahové hodnoty je její manuální určení. Manuální hodnota prahu se volí pro všechny pixely obrazu. Na základě této hodnoty se opět rozhoduje o černých a bílých pixelech výsledného obrazu.

Tato metoda samozřejmě závisí na vhodně zvoleném prahu. Velmi dobře se pracuje například s fotografií stránky s textech, kdy jsme schopni pomocí této metody dosáhnout velmi dobrého výsledku. Samozřejmě při nevhodně zvoleném prahu v obrazech s velkou hustotou informací v určitých jasových hodnotách dojde ke ztrátě informací. V obrazech, kde jsou různé lokální jasové poměry je potom velmi obtížné vůbec nějakou rozumnou hodnotu prahu najít. Různé hodnoty prahu ukazují obrázky 2.1.

(a) Práh 0,4 (b) Práh 0,5 (c) Práh 0,6

Obr. 2.1: Manuálně zvolené prahy, použito z [7]

2.1.2 Automatický práh

Opačnou metodou k manuálně zvolenému prahu je metoda automatické volby prahu. Vhodná hodnota prahu je vypočítána na základě rozložení odstínů vstupního obrazu. Jednou z metod, která funguje na tomto principu je Otsu prahování (podle svého objevitele Nobuyuki Otsu).

Metoda Otsu

Metoda Otsu zkoumá v histogramu rozložení intezit obrazu. Na základě toho je potom stanovena prahová hodnota. Je to metoda založená na statistických výpočtech. Dochází při ní k rozdělení pixelů histogramu na dvě třídy – popředí a pozadí. Zjišťuje se tak maximální mezitřídní rozptyl na jehož základě se určí prahová hodnota. Porovnání metod Otsu a Cow-Kaneko je níže na obrázku 2.2.

(a) Původní obraz (b) Otsu (c) Chow a Kaneko

Obr. 2.2: Porovnání metod Otsu a Chow-Kaneko, použito z [10]

2.1.3 Adaptivní práh

Adaptivní prahování je vlastně dynamické lokální prahování. Dojde k rozdělení obrazu na několik oblastí a v nich je následně spočítaný práh. To znamená, že není pro každý pixel obrazu stejná hodnota prahu. Výhodou prozkoumávání menších oblastní obrazu je, že v nich není tak vysoká pravděpodobnost velkých jasových rozdílů. Opět se využívá statistických metod vyhodnocení histogramů částí obrazu, z nichž se vypočítá nejvhodnější hodnota prahu.

Mezi dva základní přístupy k tomuto problému – vymezení hodnoty prahu na základě rozdělení obrazu a zjišťování prahové hodnoty pro každý pixel zvlášť.

Chow a Kaneko metoda

Chow a Kaneko metoda používá první způsob určení hodnoty prahu. Obraz je tedy rozdělen na několik překrývajících se dlaždicových oblastí. Pro všechny pixely dané oblasti je analýzou histogramu určena prahová hodnota. Tato metoda je díky své náročnosti velmi neefektivní pro zpracování obrazu v reálném čase. Viz 2.2

Wellnerova metoda

Wellnerova metoda prahování je lokální metodou prahování. To znamená, že je zjišťována hodnota prahu pro každý bod zvlášť pomocí zkoumání okolí jednotlivých bodů. Základem této metody je čtení vstupního obrazu po řádcích a rozhodování o hodnotě prahu z doposud přečtených pixelů. Vyhodnocení těchto dat závisí na charakteru daného obrazu. Můžeme použít tři možnosti postupu vyhodnocení dané množiny pixelů.

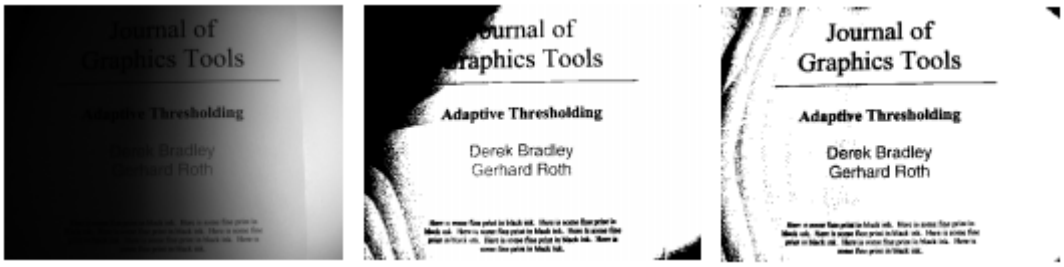
Nejjednodušší možností je stanovení prahu z průměru hodnot, další možností je využití mediánu hodnot a poslední možností je zprůměrování minimální a maximální hodnoty v množině.

Metoda je závislá na množství bodů zahrnutých do množiny výpočtu prahu. Tento vzorek bodů musí být dostatečný pro pokrytí bodů pozadí a popředí, ale zároveň nesmí být moc velký, aby rozložení jasu přestalo být uniformní.

Výhodou algoritmu je rychlost oproti Chow a Kaneko metodě. Nevýhodou jsou odlišné výsledky zpracování výsledného obrazu při průchodech opačným směrem (zleva doprava, zprava doleva). Porovnání s metodou Bradley-Roth je na obrázku 2.3.

Metoda Bradley-Roth

Metoda Dereka Bradleyho a Gerharda Rotha řeší právě problém s různými směry průchodu obrazem. Nepočítá se z hodnotami načtenými v předchozím průchodu obrazem, ale s hodnotami v čtvercovém okolí daného bodu. Docílilo se tak mnohem lepší výsledků kvality výsledného obrazu, kdy zároveň nezáleží na směru průchodu obrazem. Nevýhodou oproti Wellnerově algoritmu je pomalejší výpočet výsledného obrazu, ale stále se dá hovořit o vhodnosti použití v aplikacích běžících v reálném čase. Porovnání s metodou Wellner je na obrázku 2.3.



(a) Původní obraz (b) Wellner (c) Bradley-Roth

Obr. 2.3: Porovnání metod Wellner a Bradley-Roth, použito z [9]

2.2 Náhodný rozptyl

Náhodný rozptyl je metoda ditheringu výběru prahové hodnoty pro konkrétní bod. Tato náhodná hodnota je vybrána z intervalu $< 0, Imax >$, přičemž $Imax$ je maximální hodnota intenzity vstupního obrazu. Obecný algoritmus této metody je založen na rozhodování jestli je hodnota aktuálního pixelu vetší, nebo menší než náhodně vygenerované číslo. V případě vyšší hodnoty je přiřazena černá barva, naopak nižší hodnota zajistí barvu bílou.

Čím větší bude intenzita vstupního pixelu, tím vyšší je pravděpodobnost, že bude náhodná hodnota nižší než daná intenzita a pixelu bude přiřazena černá barva. Naopak nižší intenzita vstupu zvyšuje pravděpodobnost přiřazení bílé barvy.

Jistý problémem metody je získání opravdu náhodného čísla. K tomu se používá tzv. generátor náhodných čísel. Softwarové generátory totiž mohou vykazovat stále stejnou posloupnost náhodných čísel, na základě určité vstupní hodnoty. Proto se pro generování náhodnosti používá třeba signálů síťové karty, či čtecích hlav pevného disku.

Výsledný obraz či použití náhodného rozptylu obsahuje značné množství vysokofrekvenčního šumu, způsobeného právě pravděpodobnostním modelem. Výhodou náhodného rozptylu je schopnost věrně zobrazit nízkofrekvenční barevné přechody. V obrazech kde se vyskytují jasné přechody, či hrany objektů je však tento algoritmus neuspokojivý. Projeví se zde již zmíněná produkce šumu a původní informace v obrazu je téměř k nepoznání. Jako na obrázku 2.4.

Jednou z variant tohoto algoritmu, která se používá spíše u barevných obrazů je připočítání určité náhodné hodnoty v uniformním rozložení z daného rozmezí. Tím se obraz náhodně zašumí a až potom se použije prahování. Tato metoda funguje velmi dobře při redukci barevného obrazu. Přidáním náhodné barvy ke každému pixelu a vybráním nejvhodnější barvy z omezené palety. [13]

(a) Původní obraz (b) Náhodný rozptyl

Obr. 2.4: Náhodný rozptyl

2.3 Maticový rozptyl

V metodě maticového rozptylu se také redukuje barevný prostor na jednobitovou paletu. Pracuje se zde ale s maticí prahových hodnot, která se aplikuje na vstupní obraz. Maticový rozptyl pracuje vždy s maticí číselných hodnot, která se aplikuje po blocích na vstupní obraz. Pro každý pixel je potom provedeno prahování pomocí hodnoty v příslušném místě matice. Výsledná obraz tvoří skupiny bodů o velikosti použité matice, které vytvářejí vjem různých odstínů šedé barvy. Nejlepší výsledky poskytuje Bayerova matice tvořená 4×4 pixely.

Obecný algoritmus funguje na principu průměrování barevné hodnoty pixelů. Výsledek tohoto průměrování je brán jako původní pixel a nabývá hodnot z rozsahu odstínů barev. Pokud je původní pixel stejné nebo nižší hodnoty, bude v dané matici bílý, pokud je vyšší bude černý.

Výhodou této metody je, že matice může být transponována či otáčena bez omezení na funkčnosti algoritmu. Bayerova matice se stala nejpoužívanější, protože je její rozměr je mocninou dvou, což jak Bayer ukázal se jeví jako nejlepší matice pro uspořádaný dithering. Samozřejmě si můžeme nadefinovat svoje vlastní matice hodnot prahů, některé se budou hodit pro určité specifické úpravy obrazu. [13] Příklady matic jsou na obrázku 2.5.

(a) Matice 2×2 (b) Matice 4×4 (c) Matice 8×8

Obr. 2.5: Přehled Bayerových matic

2.4 Distribuce zaokrouhlovací chyby

Všechny metody ditheringu obrazu se snaží omezit ztrátu informací vznikající redukcí barevného prostoru na jednobitovou paletu. Minimalizace této ztráty se dosahuje náhodným rozptylem, či maticovým rozptylem v pravidelných obrazcích. Tato metoda je také založena na prahování, ovšem rozdílem je reakce na možnou chybu při stanovení prahové hodnoty. Vyčísluje se ztracená informace a obraz je dále upravován, aby tuto ztrátu kompenzoval.

Chybou je myšlena hodnota rozdílu intenzity původního pixelu a hodnota intenzity výstupního pixelu. Chyba se vypočítá z intenzity vstupního pixelu, například 0 a hodnoty výstupního pixelů daného prahováním. Při této kombinaci jsme se dopustili chyby o hodnotě 100. Tato chyba je potom distribuována na okolní pixely obrazu, což zajistí kompenzaci chyby u dalších zpracovávaných pixelů.

Existuje několik možností přenášení informací o chybě na okolní pixely obrazu. Obraz je u těchto metod procházen sekvenčně, tedy po řádcích zleva doprava. Distribuce chyby se potom přenáší na pixely následující ve směru čtení a na pixely pod právě zpracovávaným pixelem (zde ve směru jak dopředném, tak i zpětném) – vše na základě použité metody. Chyba je hodnota přičtená jako korekce o další zpracovávaným pixelů. Čím více pixelů z okolí je do distribuce chyby zapojeno, tím kvalitnější je výsledek.

Jednotlivé níže popsané metody se liší právě přenášením chyby na okolní body. Tyto metody používají matici hodnot určující poměr rozložení chyby mezi body. Nejjednodušší metodou je distribuce, při které je obraz prahován hodnotou 0,5. Chyba je šířena 100% na další pixel. I pro jednoduchost této metody podává celkem dobré výsledky. Nevýhodou je možnost vytváření nežádoucích artefaktů v obraze. [13]

(a) Původní obraz (b) Distribuce chyby

Obr. 2.6: Práhování 0,5 a distribuce 100% chyby, použito z [7]

2.4.1 Floyd-Steinberg

Metoda Floyd-Steinberg je nejznámější metodou rozptylu s distribucí chyby. Pro distribuci chyby se používá matice koeficientů zlomků se jmenovatelem 16. Konkrétní koeficienty jsou 7/16 pro přenos chyby na další pixel a pro přenos na další řádek jsou to koeficienty 3/16, 5/16 a 1/16, viz obrázek 2.7. Čtení vstupního obrazu opět probíhá po řádcích, v případě opačného směru procházení stačí matici hodnot otočit.

Algoritmus přičítání chyby Floyd-Steinberg je používán především pro dobré výsledky a malou výpočetní náročnost (dělení 16 je bitová operace posunu – dělení mocninou čísla 2).

Existuje je jedna varianty Floyd- Steinbergova algoritmu, která je zjednodušení oproti předchozímu. Matice koeficientů má ve jmenovateli číslo 8 a přenášené koeficienty jsou jen tři – 3/8 pro další pixel, dále 3/8 a 2/8 pro další řádek. Tato metoda samozřejmě neposkytuje tak dobré výsledky jako původní verze, ale poskytuje dobré výsledky při procházení sudých řádků opačným směrem než liché řádky. [7, 4]Výsledek je patrný z obrázku 2.8.

		$\frac{7}{16}$
$\frac{3}{16}$	$\frac{5}{16}$	$\frac{1}{16}$

Obr. 2.7: Matice koeficientů Floyd-Steinberg

(a) Původní obraz (b) Floyd-Steinberg

Obr. 2.8: Metoda Floyd-Steinberg, použito z [7]

2.4.2 Jarvis, Judice a Ninke

Autory tohoto algoritmu jsou pánové F. Jarvis, C. N. Judice a W. H. Ninke. Jejich algoritmus odtraňuje omezení alogritmu Floyd-Steinberg v distribuci chyby pouze na čtyři sousední pixely. Jejich algoritmus předává chybu na dvanáct sousedních pixelů, podle následující matice: 2.9.

			$\frac{7}{48}$	$\frac{5}{48}$
$\frac{1}{16}$	$\frac{5}{48}$	$\frac{7}{48}$	$\frac{5}{48}$	$\frac{1}{16}$
$\frac{1}{48}$	$\frac{1}{16}$	$\frac{5}{48}$	$\frac{1}{16}$	$\frac{1}{48}$

Obr. 2.9: Matice koeficientů Jarvis, Judice a Ninke

Metoda tak poskytuje lepší výsledky v zobrazení výsledného obrazu, avšak logicky klade vyšší nároky na výpočetní výkon. Distribuce chyby se přenáší na větší okolí a dělení s jmenovatelem 48 už není realizovatelné pomocí mocniny dvou. [7] [4]

(a) Původní obraz (b) Jarvis, Judice a Ninke

Obr. 2.10: Metoda Jarvis, Judice a Ninke, použito z [7]

2.4.3 Stucki

Metoda stucki je obměnou předchozí metody Jarvis, Judis a Ninke. Distribuce do okolních bodů obrazu zůstává stejná, mění se ale koeficienty matice. Jiné koeficienty matice poskytují výhodu v redukci operace dělení. Vypočítá se první hodnota a všechny další koeficienty už jsou jen bitový posun. [7] [4]

			$\frac{4}{21}$	$\frac{2}{21}$
$\frac{1}{21}$	$\frac{2}{21}$	$\frac{4}{21}$	$\frac{2}{21}$	$\frac{1}{21}$
$\frac{1}{42}$	$\frac{1}{21}$	$\frac{2}{21}$	$\frac{1}{21}$	$\frac{1}{42}$

Obr. 2.11: Matice koeficientů Stucki

(a) Původní obraz (b) Stucki

Obr. 2.12: Metoda Stucki, použito z [7]

2.4.4 Burkes

Ještě větší optimalizaci výpočtu přináší metoda Burkes. Je odstraněn poslední řádek z metody Stucki. Dále přináší výhodu výpočtů bitovým posunem, při koeficientu s jmenovatelem 32. [7] [4]

			$\frac{1}{4}$	$\frac{1}{8}$
$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$

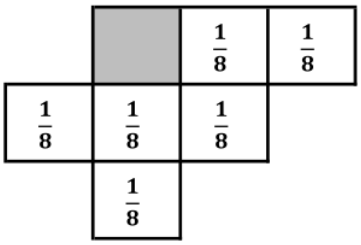
Obr. 2.13: Matice koeficientů Burkes

(a) Původní obraz (b) Burkes

Obr. 2.14: Metoda Burkes, použito z [7]

2.4.5 Atkinson

Další metodou distribuce chyby je metoda Atkinson. Je to metoda podobná Jarvis, Judice a Ninke, s rozdílem rychlosti výpočtu. Hlavním znakem je distrubuce pouze 75% chyby. To má za následek ztrátu informací v krajích histogramu, tedy ztrátu světlých a tmavých míst obrazu (ztrátu kontrastu). Dobré výsledky však zaznamenává ve středních tónech. [7] [4]



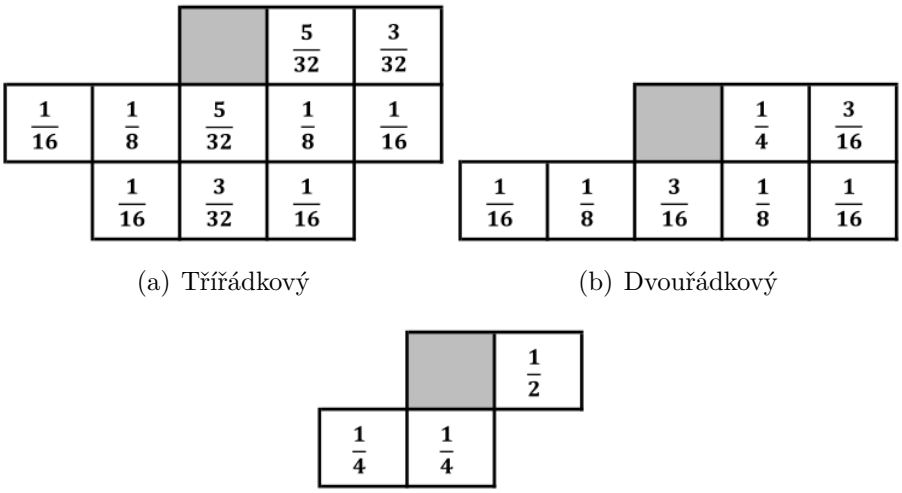
Obr. 2.15: Matice koeficientů Atkinson

(a) Původní obraz (b) Atkinson

Obr. 2.16: Metoda Atkinson, použito z [7]

2.4.6 Sierra

Metoda distribuce chyb Sierra nabízí hned tři matice koeficientů. Jedná se obdoby matic předchozích metod. Třířádková matice je obdobou matice Jarvis, Judis a Ninke s vynecháním některých koeficientů třetího řádku, což má za následek zvýšení rychlosti zpracování. Dvouřádková matice je zjednodušení matice třířádkové se změnou několika koeficientů, opět z důvodů rychlosti výpočtu. Poslední nejjednodušší Sierra maticí, je zjednodušená matice Floyd-Stenbergovi metody, kterou nazýváme "Filter Lite". [7] [4]



(a) Třířádkový (b) Dvouřádkový

(c) Filter Lite

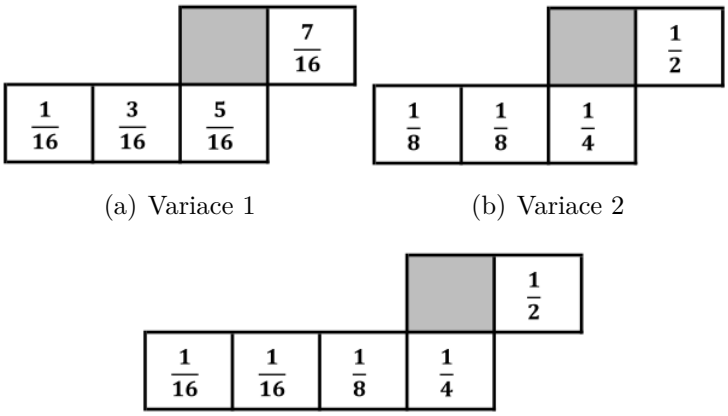
Obr. 2.17: Matice koeficientů Sierra

(a) Třířádkový (b) Dvouřádkový (c) Filter Lite

Obr. 2.18: Přehled metod Sierra, použito z [7]

2.4.7 Shiau-Fan

Metoda Shiau-Fan je opět obdobou metody Floyd-Steinberg. Používá pouze upravené matice a jiné koeficienty. Hlavním přínosem této metody má být lepší odstranění artefaktů oproti původní metodě Floyd-Steinberg. [7] [4]



(a) Variace 1 (b) Variace 2

(c) Variace 3

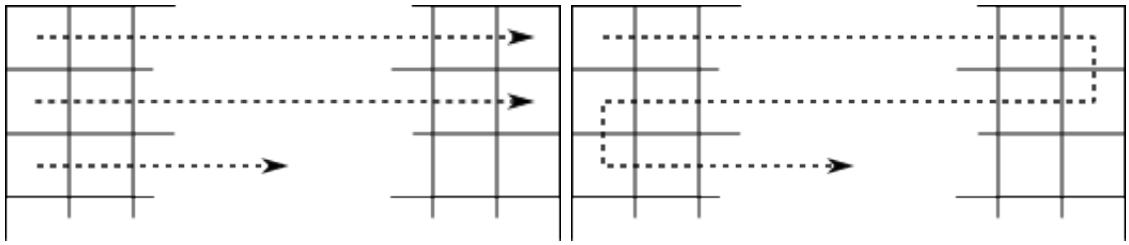
Obr. 2.19: Matice koeficientů Shiau-Fan

(a) Variace 1 (b) Variace 2 (c) Variace 3

Obr. 2.20: Přehled metod Shiau-Fan, použito z [7]

2.5 Směr analýzy pixelů

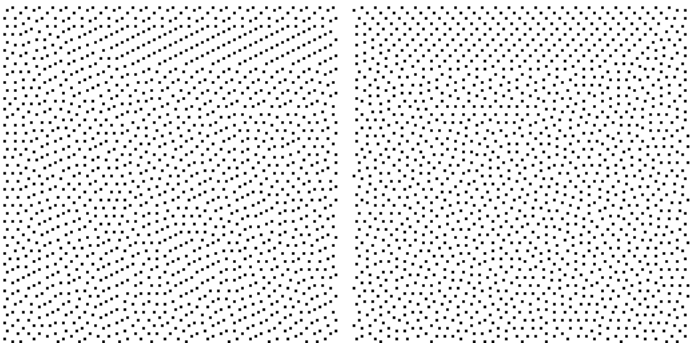
Výsledný obraz je při ditheringu ovlivněný směrem čtení a následného zpracování pixelů. Základní metoda je čtení pixelů zleva doprava, kdy se na konci řádku přeskočí na řádek další a pokračuje se stejným směrem. Druhou metodou je čtení následující ho řádku opačným směrem, tedy že liché řádky se čtou směrem zleva doprava a sudé zprava doleva, tak jak je ukázáno na obrázku 2.21. [7]



(a) Zleva doprava (b) Opačný po řádcích

Obr. 2.21: Směry zpracování pixelů, použito z [7]

Na metody prahování a náhodného rozptylu nemá změna směru čtení pixelů zásadní vliv. Projeví se spíše jen u metody Wellner. To ovšem neplatí pro metody distribuce chyby, kdy se na další pixely přenáší informace o chybě. U těchto metod se potom přenášená chyba na konci řádku přenáší na první pixel řádku následujícího.



(a) Zleva doprava (b) Opačný po řádcích

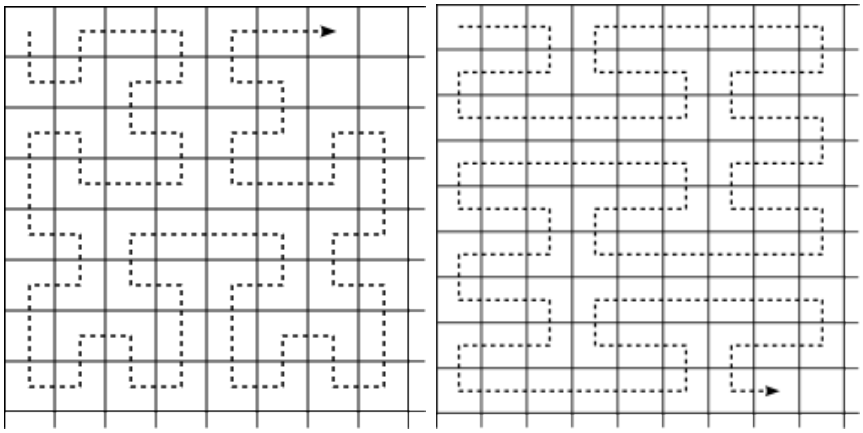
Obr. 2.22: Artefakty při různém směru zpracování pixelů, použito z [7]

Cílem opačného čtení řádku je omezení vzniku artefaktů v obraze. Jak je vidět na obrázcích 2.22 a 2.23. Proto se využívá opačného čtení následujících řádků. Ovšem to zase může způsobovat vznik nových artefaktů. Při použití metody Floyd-Steinberg je rozdíl v různých směrech čtení velmi malý.

Mezi další způsoby čtení pixelů v obraze patří Hilberotva a Peanova křivka 2.24. U metody Floyd-Steinberg jsou výsledky opět téměř neznatelné, viz 2.25. [7]

(a) Zleva doprava (b) Opačný po řádcích

Obr. 2.23: Směry zpracování pixelů Floyd-Steinberg, použito z [7]



(a) Hilbertova (b) Peanova

Obr. 2.24: Směry zpracování pixelů pomocí křivek, použito z [7]

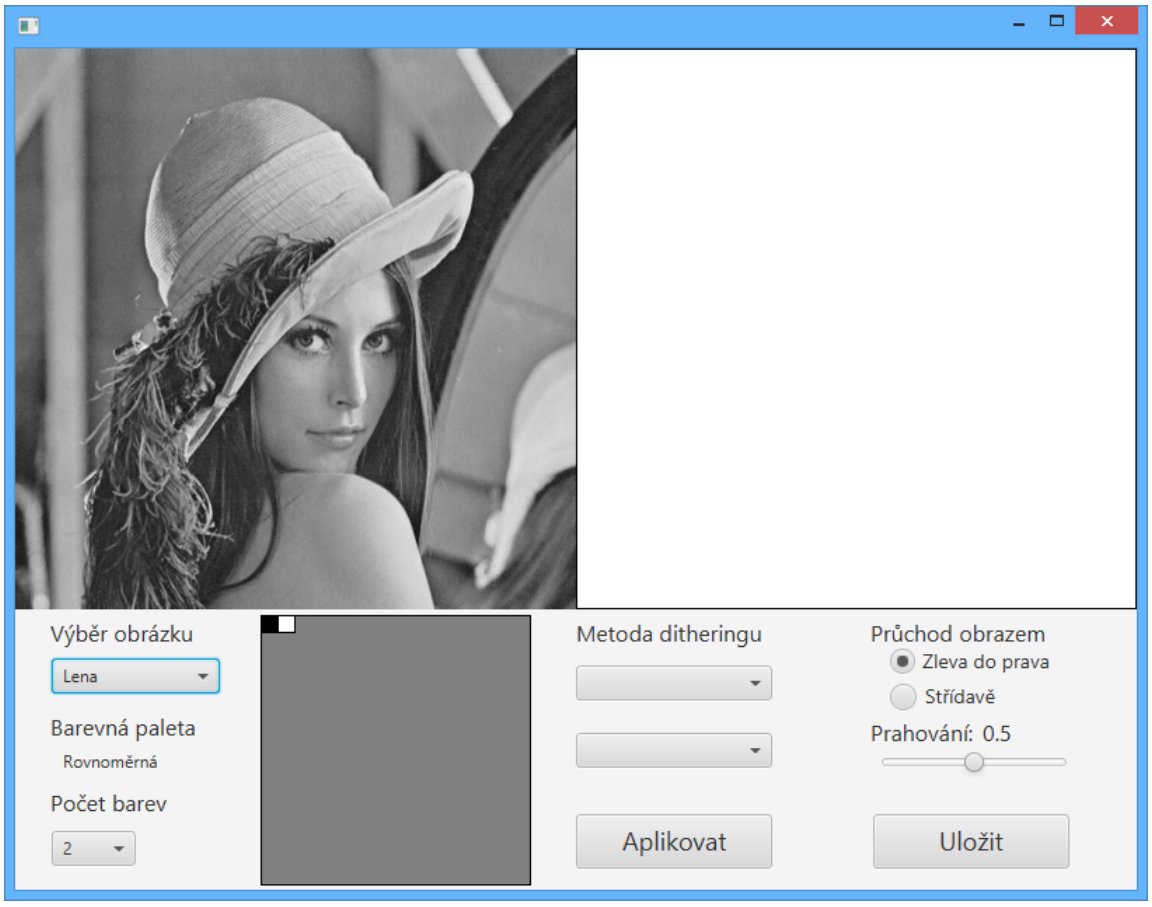
(a) Hitbertova (b) Peanova

Obr. 2.25: Metoda Floyd-Steinberg pomocí křivek, použito z [7]

3 PROGRAMOVÁ ČÁST

3.1 Grafický návrh Java-appletu

Základem appletu bude načtení a zobrazení původního obrazu. Dále možnost nastavení barevné hloubky výstupního obrazu, barevné palety, způsob průchodu obrazem. Samozřejmě také použitá metoda ditheringu vybíraná ve dvou krocích. V prvním kroku je to, zda se jedná o prahování, náhodný rozptyl, či distribuci chyby. V dalším kroku výběr metody spadající pod vybraný typ ditheringu. Některé metody budou mít ještě dílčí nastavení, například nastavení hodnoty prahu při prahování. V neposlední řadě možnost aplikovat vybrané změny na obrázek a možnost uložit jej do souboru.



Obr. 3.1: Návrh vzhledu java-appletu

Na obrázku 3.1 je vidět finální vzhled java-appletu. Je zde vidět umístění dvou základní prvků, tedy vstupního obrázku vlevo a výstupního vpravo. Výběr ze šesti obrázků, palety, počtu barev. Čtverec zobrazující použité barvy v obraze včetně jejich počtu. Dále průchod obrazem, možnosti ditheringu a výběr dané metody.

Obrázek se generuje na základě nastavení jednotlivých položek tlačítkem Aplikovat. Funkční je uložení obrázku do souboru pomocí tlačítka Uložit.

Graficky je applet navržený v JavaFX Scene Builder 2.0. Pro JavaFX bohužel nic jiného nezbývá. Je smutné, že nelze udělat grafický návrh v GUI editoru programu NetBeans, který je dle mého názoru kvalitou nedostižný. V programu Scene Builder totiž zapomněli na několik zásadních možností nastavení, které se musí složitě dopisovat kódem. Mezi tyto nedostatky patří například dopisování jednotlivých položek výběrů v ChoiceBoxu.

3.2 Programování

3.2.1 Technické informace

Základní požadavkem pro program byl programovací jazyk Java. Dále se mělo jednat o java-applet. Jako nejvhodnější pro splnění těchto požadavků bylo zvoleno JavaFX, právě kvůli jednoduchosti tvorby webového java-appletu a předchozích znalostí.

Jako vývojové prostředí jsem zvolil NetBeans IDE ve verzi 8.0. Toto prostředí by bylo nefunkční bez Java JRE a Java JDK ve verzi 8.0 update 05. Grafický návrh obstarává JavaFX Scene Builder ve verzi 2.0. Implementace JavaFX již není nutné řešit, je součástí zmíněného JRE a JDK.

3.2.2 Vývoj programu

Základem vývoje bylo zprovoznění všech grafických prvků. Tedy zajištění správných výběrů ChoiceBoxu, popisků tlačítek, textů, spárování komponenty RadioButton a další.

Následovala inicializace 6 obrázků, které byly voleny s ohledem na možné zajímavé výsledky generované metodami 4.1. Načtení obrázku do komponenty ImageView originálního obrázku. Pro výstup je použita komponenta ImageViewOutput, ve které je zobrazen vygenerovaný obrázek. Naprosto zásadní je zobrazení obrázků 1:1. Pokud by docházelo k nějaké transformaci obrázku (například zvětšení, či zmenšení), bude do obrázku zanesena chyba přepočtu obrazu grafickou kartou. Uvedené obrázky jsou samozřejmě přizpůsobeny velikostně tak, aby k tomuto problému nedocházelo. Pokud by byly do programu vloženy jiné obrázky, vše bude fungovat, ale správně zobrazený bude až uložený obrázek, ten bude 1:1 v jakékoliv velikosti.

V programu jsou použity obrázky v odstínech šedi, nebylo tedy nutné řešit převod barevného obrázku na černobílý. Bylo ale nutné řešit omezení počtu barev v rovnoměrné paletě barev. Řešení je převod pixelů na hodnotu double v rozmezí 0,0 až

1,0. Dále už je jen pomocí cyklu přidáno rozdělení pixelů do několika úrovní podle počtu zvolených barev.

Následovala implementace jednotlivých metod. Základní průchod obrazem je vždy z levého horního rohu do pravého spodního. Obraz procházíme po řádcích. V případě, že narazíme na konec obrazu, přeskočí algoritmus na nový řádek.

Pro manuální prahování 2.1.1byla implementace snadná, ovlivňujeme pouze hodnotu, od které bude daný pixel bílý nebo černý. Automatické prahování Otsu 2.1.2 už je poměrně složitější algoritmus, který počítá průměrnou hodnotu pixelu a na základě toho rozhodne o hranici prahu. Náhodný rozptyl je založený na pseudonáhodnosti, jak bude výslední pixel barevný.

Maticový rozptyl 2.3 obsahuje možnost Bayerovi matice o 2×2 prvcích s hodnotami 0, 7, 11, 4. Tato matice se uplatňuje při průchodu obrazem a ovlivňuje výslednou podobu nového pixelu.

Metody distribuce chyby jsou založeny na stejném algoritmu. Mění se pouze velikost matice zanášených chyb. Nejjednodušší matici má metoda Floyd-Steinberg 2.4.1. Zanáší se pouze 4 chyby. Ty jsou uložené do dočasných proměnných temp. Jednotlivé chyby se potom zanesou do výsledných pixelů a pokračuje se na další pixel. Podle volby možnosti průchodu obrazem se potom algoritmus zanáší chyby na pixely v určitém směru. Algoritmus má dva vnořené cykly pro postup na vodorovné ose X a svislé ose Y. Kódy některých metod jsou v příloze C.

Zapracovány jsou tedy metody prahování 2.1, jak automatické, tak manuální. Metody náhodného rozptylu 2.2 a maticového rozptylu s maticí 2×2 prvků 2.3. Z metod Distribuce chyby jsou zapracovány: Floyd-Steinberg 2.7, Jarvis 2.9, Stucki 2.11, Burkes 2.13, Atkinson 2.15, třířádková Sierra 2.17, a Shiau- Fan variace 3 2.19.

Předposlední krok obsahoval ošetření výjimek. Bylo potřeba ošetřit, kdy se jaká komponenta má zobrazovat. Pro prahování je potřeba zviditelnit komponentu Slider pro možnost volby manuálního prahu. Zároveň tato metoda funguje jen pro dvě barvy, musí tedy zmizet možnost volby barev a barevná paleta se přepne do dvou barev. Pro metody distribuce chyby bylo potřeba přidat zviditelnění možností vodorovného a střídavého průchodu obrazem.

Na úplný závěr byla zvolena základní volba jednotlivých komponent, původního obrázku a počtu barev. Nepůsobilo dobře, když byl applet při spuštění prázdný.

4 SROVNÁNÍ JEDNOTLIVÝCH METOD

Tato kapitola subjektivně srovnává výsledky jednotlivých metod. Zařazeno je subjektivní srovnání pomocí dotazníku a vlastní subjektivní srovnání.

Do srovnání byly zařazeny obrázky z naprogramovaného appletu. Byly zvoleny s ohledem na zajímavé možnosti vygenerovaných obrázků jednotlivými metodami. Jednalo se o obrázky z výběru níže 4.1.

Obrázek Lena je referenční, srovnáván je snad ve všech publikacích, nesmí tedy chybět ani v této práci. Obrázek Mříž byl zvolen kvůli pravidelnosti tvarů objevujících se v obraze. V obrázku Kaktus bude zajímavé sledovat zachování detailů chlupů rostoucích z kaktusu. Obrázek s názvem Houby byl zvolen kvůli jehličí nacházejícího se v pozadí. V obrázku Krajina bude zajímavé sledovat rozlišení jednotlivých stromů a vykreslení oblohy. Poslední obrázek s názvem Zvíře byl zvolen pro stejnou barvu zvířat a pozadí s ohledem na to, jestli vůbec v jednotlivých metodách zvířata nesplynou s pozadím.

4.1 Subjektivní srovnání – dotazník

Důležitou volbou bylo, v jakém systému dotazník vytvořit. Volba padla nakonec na Google formulář. Vytváření dotazníku je jednoduché, přehledné, navíc zdarma. Nabízelo se vytvořit dotazník v jiném webovém formuláři, ale pro tento účel použitelné dotazníky byly placené. Možnost vytvořit dotazník v programu Excel byla zavržena, pro velmi špatnou možnost distribuce dotazníku k respondentům. Distribuce webového formuláře Google je velmi jednoduchá. Formulář také umožňuje dobrý výstup dat, generuje i výsledné srovnávací grafy. Ty bohužel nejsou pro tuto práci dostatečné. Byly tedy využity pouze záznamy jednotlivých odpovědí z vygenerované tabulky. Srovnání pomocí dotazníku přineslo hned několik problémů, které bylo potřeba vyřešit.

Zásadní problém byla práce Google formuláře s obrázky. Formulář je sice vytvářen pro obrázky do šířky 760 pixelů, ale výsledný formulář, který vidí respondenti, má velikost šíře obrázků pouze 586 pixelů. Aby byly jednotlivé metody srovnatelné, musí být všechny obrázky zobrazené 1:1. To je naprosto zásadní. Pokud by docházelo k nějaké transformaci obrázku (například zvětšení, či zmenšení), bude do obrázku zanesena chyba přepočtu obrazu grafickou kartou. Obrázky jsou samozřejmě přizpůsobeny velikostně tak, aby k tomuto problému nedocházelo.

Otázky byly zvoleny dvě. První otázkou je podobnost výsledného obrázku originálem. Druhou otázkou je jak na respondenta obrázek působí. Obrázky jsou generovány pouze ve dvou barvách, jsou tedy černobílé. U černobílých obrázků jsou

(a) Lena

(b) Houby

(c) Mříž

(d) Kaktus

(e) Krajina

(f) Zvíře

Obr. 4.1: Obrázky vybrané do appletu

nejlépe vidět jednotlivé rozdíly jednotlivých metod.

Dalším problémem byla délka dotazníku. Výslednou délku objasňuje rovnice níže4.1. Základem je počet obrázků násobený počtem zobrazovacích metod. Hodnota +1 je přičtena pro originální obrázek. Průchod jednou stránkou a zakliknutí

odpovědí na dvě otázky bylo změřeno na 15 sekund. Následujících 60 sekund je přičteno pro první a závěrečnou stránku.

$$Délka\ dotazníku = počet\ obrázků \times (počet\ metod + 1) \times 15 + 60\ [s] \tag{4.1}$$

Stanoveným maximem byla délka vyplňování 8 minut. Je to doba, po kterou ještě respondenti udrží pozornost a vyplní celý dotazník. To ovšem obnášelo vypuštění jednotlivých metod i některých obrázků.

Vypuštěna byla metoda automatického prahování. Tato metoda vykazovala ve zvolených obrázcích hodnoty prahu kolem 0,45. Prahování tedy stačilo zvolit manuální s hodnotou 0,5. Dále byla vynechána metoda Jarvis pro podobnost s metodami Stucki a Sierra třířádková. To umožnilo generování matice 10 obrázků pro náhled výsledků metody.

Zapracovány jsou tedy metody prahování 2.1, jak automatické, tak manuální. Metody náhodného rozptylu 2.2 a maticového rozptylu s maticí 2×2 prvků 2.3. Z metod Distribuce chyby jsou zapracovány: Floyd-Steinberg 2.7, Stucki 2.11, Burkes 2.13, Atkinson 2.15, třířádková Sierra 2.17, a Shiau- Fan variace 3 2.19. Ukázka porovnávaného obrázku 4.2, nejprve originální obrázek, potom jmenované metody.

Počet obrázků bylo nutné omezit na 3, zvoleny byly obrázky Lena, Houby a Krajina. Počet metod je 9 přičemž připočítáváme i originální obrázek. Výsledná rovnice je potom 4.2:

$$Délka\ dotazníku = 3 \times (9 + 1) \times 15 + 60 = 510\ [s]. \tag{4.2}$$

Výsledek 510 sekund odpovídá 8,5 minutám vyplňování dotazníku. Není to sice tolik jako zvolený cíle, ale už nebylo možné z dotazníku cokoliv vypustit. Nezbývalo než se s touto hodnotou spokojit.

Obr. 4.2: Lena obrázky pro dotazník

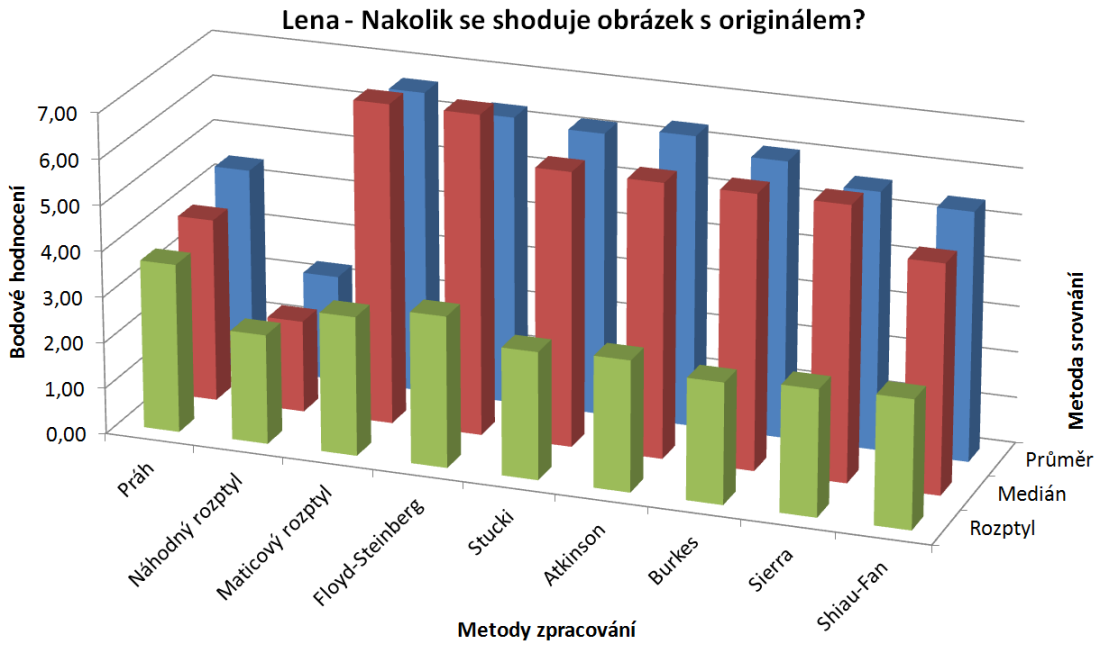
4.1.1 Výsledky srovnání

Počet respondentů, kteří vyplnili dotazník, se vyšplhal na 57. Z těchto odpovědí budou vycházet následná srovnání zanesená do tabulek a grafů. Výsledky průzkumu byly zaneseny do šesti tabulek. V prvních dvou jsou zobrazeny výsledky jednotlivých metod a obrázků. Výsledky jsou v podobě průměru, mediánu a rozptylu jednotlivých odpovědí.

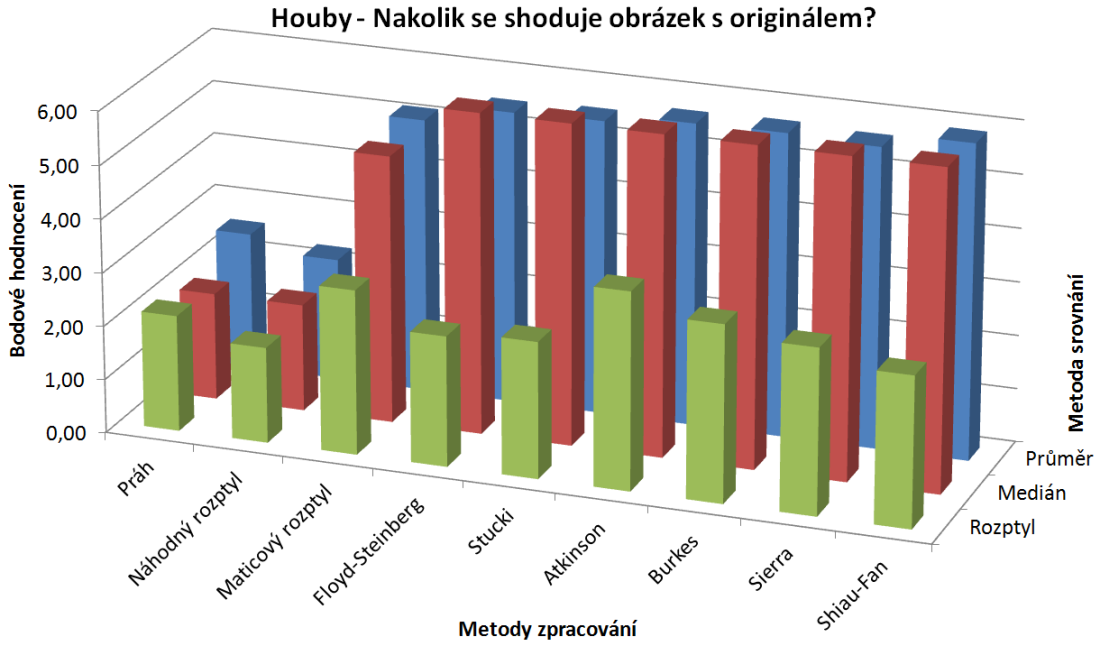
Tab. 4.1: Souhrn odpovědí na první otázku

Nakolik se shoduje obrázek s originálem?				
Obrázek	**Metoda**	**Průměr**	**Medián**	**Rozptyl**
Lena	Práh	4,46	4	3,69
	Náhodný rozptyl	2,32	2	2,39
	Maticový rozptyl	6,63	7	3,04
	Floyd-Steinberg	6,32	7	3,30
	Stucki	6,19	6	2,79
	Atkinson	6,37	6	2,86
	Burkes	6,05	6	2,65
	Sierra	5,63	6	2,76
	Shiau-Fan	5,44	5	2,81
Houby	Práh	2,57	2	2,17
	Náhodný rozptyl	2,28	2	1,79
	Maticový rozptyl	5,13	5	3,08
	Floyd-Steinberg	5,47	6	2,44
	Stucki	5,51	6	2,55
	Atkinson	5,66	6	3,70
	Burkes	5,68	6	3,31
	Sierra	5,62	6	3,10
	Shiau-Fan	5,89	6	2,82
Krajina	Práh	2,98	3	2,76
	Náhodný rozptyl	1,76	1	1,59
	Maticový rozptyl	4,94	5	3,10
	Floyd-Steinberg	5,82	6	3,01
	Stucki	5,65	6	2,96
	Atkinson	5,54	6	2,98
	Burkes	5,60	5	2,97
	Sierra	5,25	5	3,00
	Shiau-Fan	5,42	6	3,32

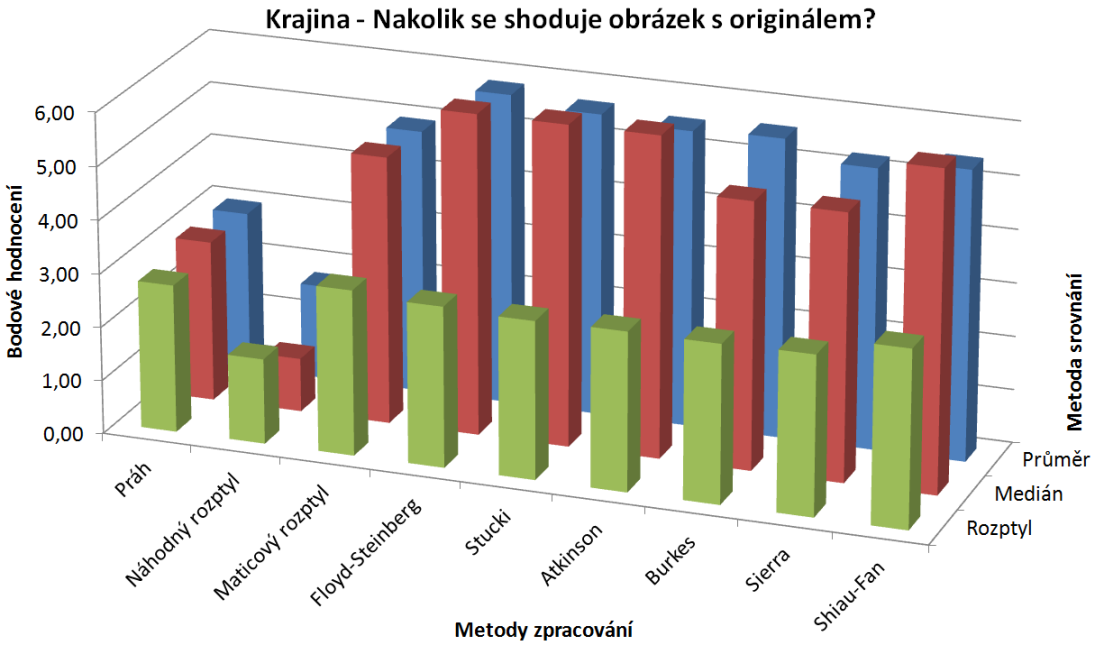
Tabulka 4.1 ukazuje, jak dopadly jednotlivé metody a jednotlivé obrázky v hodnocení pro první otázku (Nakolik se shoduje obrázek s originálem?). Celkové výsledky lépe ukazují výsledné grafy 4.3.



Obr. 4.3: Lena – Nakolik se shoduje obrázek s originálem?



Obr. 4.4: Houby – Nakolik se shoduje obrázek s originálem?

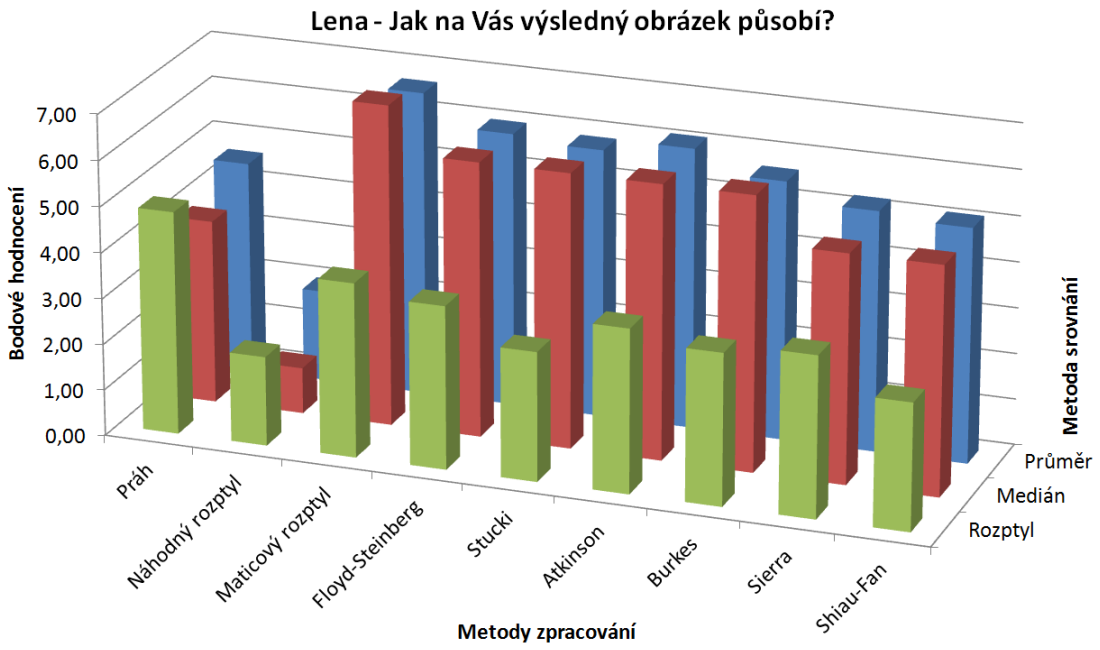


Obr. 4.5: Krajina – Nakolik se shoduje obrázek s originálem?

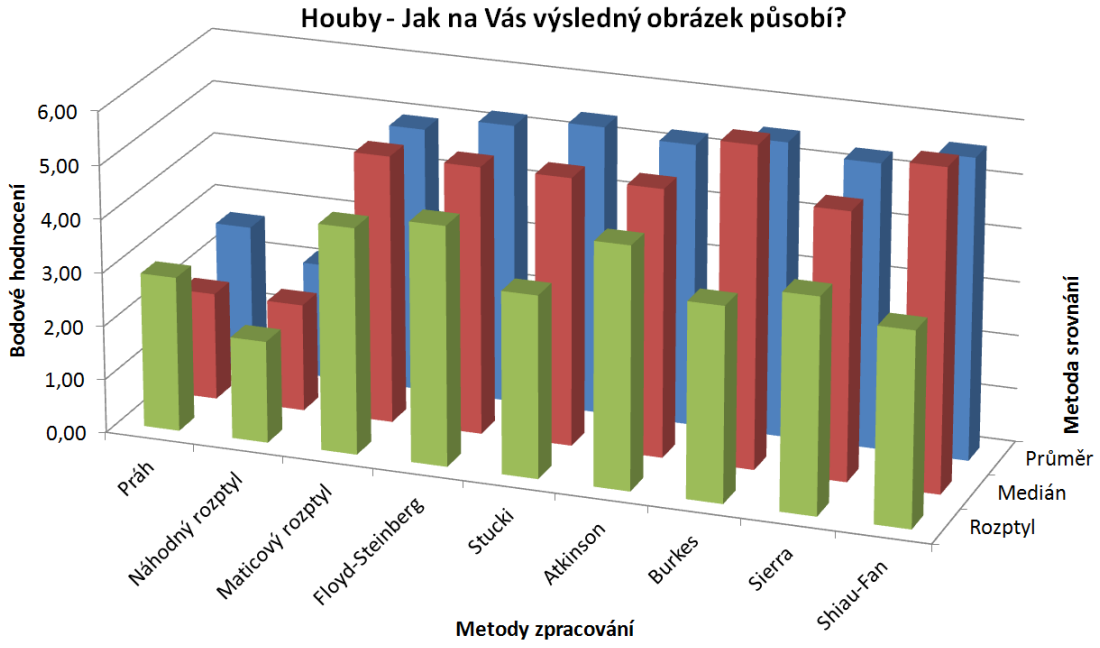
Tab. 4.2: Souhrn odpovědí na druhou otázku

Jak na Vás výsledný obrázek působí?				
Obrázek	**Metoda**	**Průměr**	**Medián**	**Rozptyl**
Lena	Práh	4,63	4	4,86
	Náhodný rozptyl	1,98	1	1,95
	Maticový rozptyl	6,65	7	3,81
	Floyd-Steinberg	5,98	6	3,56
	Stucki	5,86	6	2,82
	Atkinson	6,12	6	3,58
	Burkes	5,65	6	3,32
	Sierra	5,25	5	3,52
	Shiau-Fan	5,12	5	2,77
Houby	Práh	2,70	2	2,89
	Náhodný rozptyl	2,15	2	1,90
	Maticový rozptyl	4,96	5	4,23
	Floyd-Steinberg	5,23	5	4,48
	Stucki	5,40	5	3,41
	Atkinson	5,26	5	4,53
	Burkes	5,51	6	3,65
	Sierra	5,32	5	4,03
	Shiau-Fan	5,62	6	3,63
Krajina	Práh	4,18	3	7,95
	Náhodný rozptyl	1,73	1	1,41
	Maticový rozptyl	5,20	5	4,59
	Floyd-Steinberg	5,33	5	3,53
	Stucki	5,42	5	3,05
	Atkinson	5,46	6	4,33
	Burkes	5,40	5	3,82
	Sierra	4,92	5	3,57
	Shiau-Fan	5,25	5	3,73

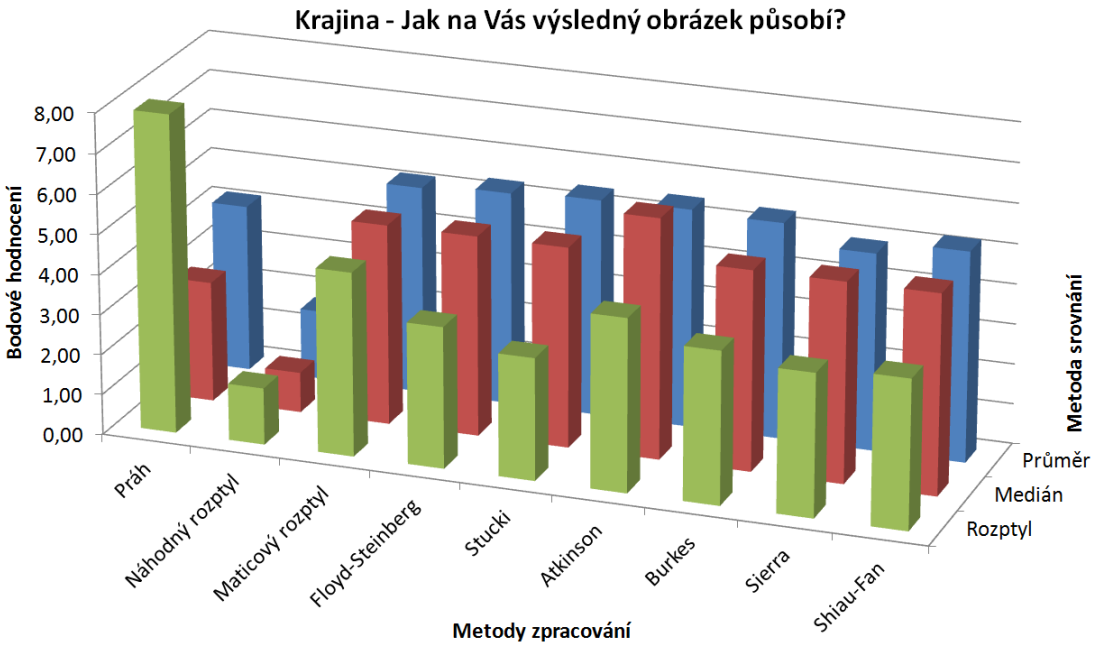
Tabulka 4.2 ukazuje jak dopadly jednotlivé metody a jednotlivé obrázky v hodnocení pro druhou otázku (Jak na Vás výsledný obrázek působí?). Výsledky lze vyčíst také z grafů 4.6.



Obr. 4.6: Lena – Jak na Vás výsledný obrázek působí?



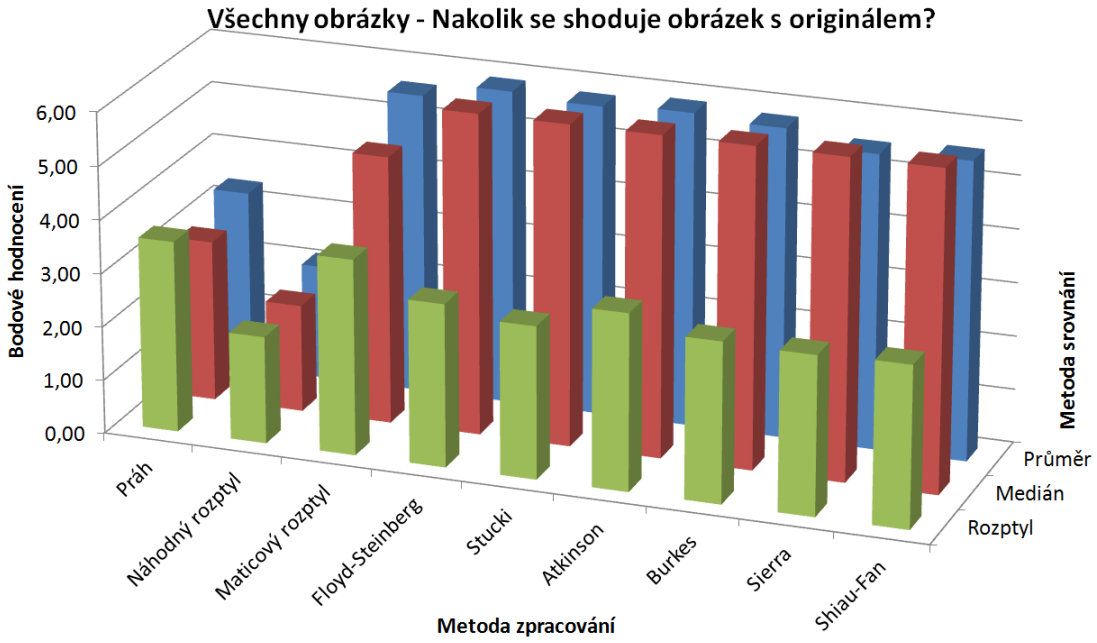
Obr. 4.7: Houby – Jak na Vás výsledný obrázek působí?



Obr. 4.8: Krajina – Jak na Vás výsledný obrázek působí?

Tab. 4.3: Celkové porovnání první otázky pro všechny obrázky

Nakolik se shoduje obrázek s originálem?				
Obrázek	**Metoda**	**Průměr**	**Medián**	**Rozptyl**
Všechny	Práh	3,37	3	3,57
	Náhodný rozptyl	2,13	2	2,00
	Maticový rozptyl	5,61	5	3,66
	Floyd-Steinberg	5,88	6	3,05
	Stucki	5,80	6	2,85
	Atkinson	5,87	6	3,31
	Burkes	5,78	6	3,01
	Sierra	5,51	6	2,98
	Shiau-Fan	5,58	6	3,02

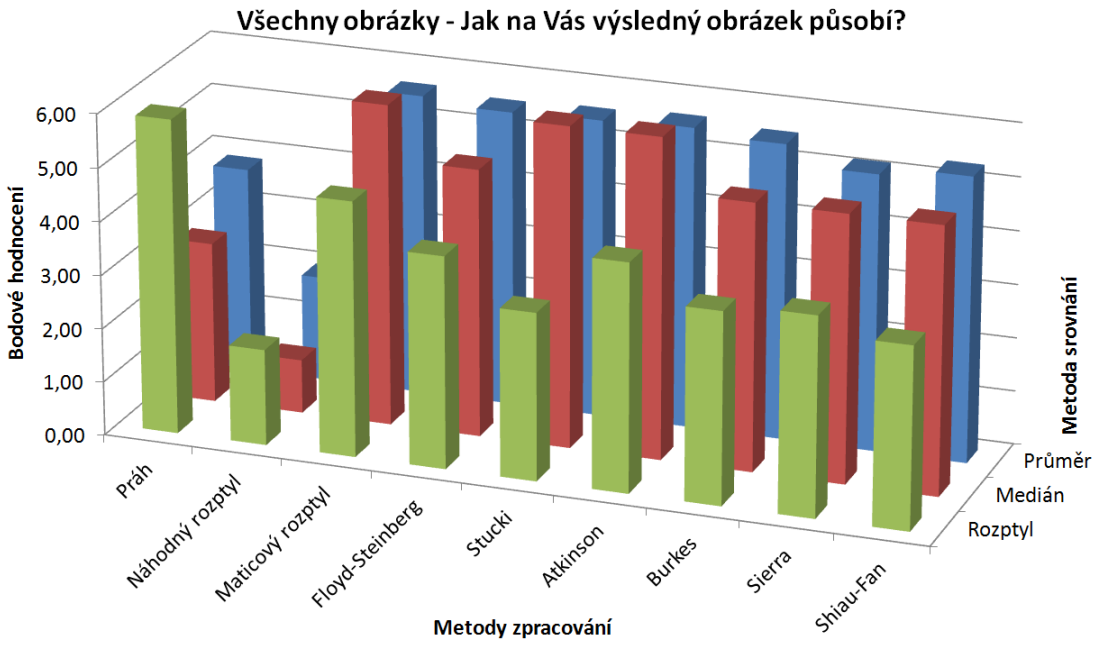


Obr. 4.9: Všechny obrázky – Nakolik se shoduje obrázek s originálem?

Z celkového porovnání první otázky je vidět 4.3 4.9, že nejvíce se shoduje s obrazem Metoda Floyd-Steinberg. Zajímavý je i stejný medián u všech metod distribuce chyby. Rozptyl odpovědí je nejvyšší pro Maticový rozptyl, nejmenší pro Náhodný rozptyl. Metoda náhodného rozptylu dopadla nejhůře, co se týče průměru z odpovědí.

Tab. 4.4: Celkové porovnání druhé otázky pro všechny obrázky

Jak na Vás výsledný obrázek působí?				
Obrázek	**Metoda**	**Průměr**	**Medián**	**Rozptyl**
Všechny	Práh	3,85	3	5,88
	Náhodný rozptyl	1,96	1	1,79
	Maticový rozptyl	5,64	6	4,77
	Floyd-Steinberg	5,52	5	3,97
	Stucki	5,57	6	3,13
	Atkinson	5,63	6	4,27
	Burkes	5,52	5	3,60
	Sierra	5,17	5	3,73
	Shiau-Fan	5,33	5	3,41

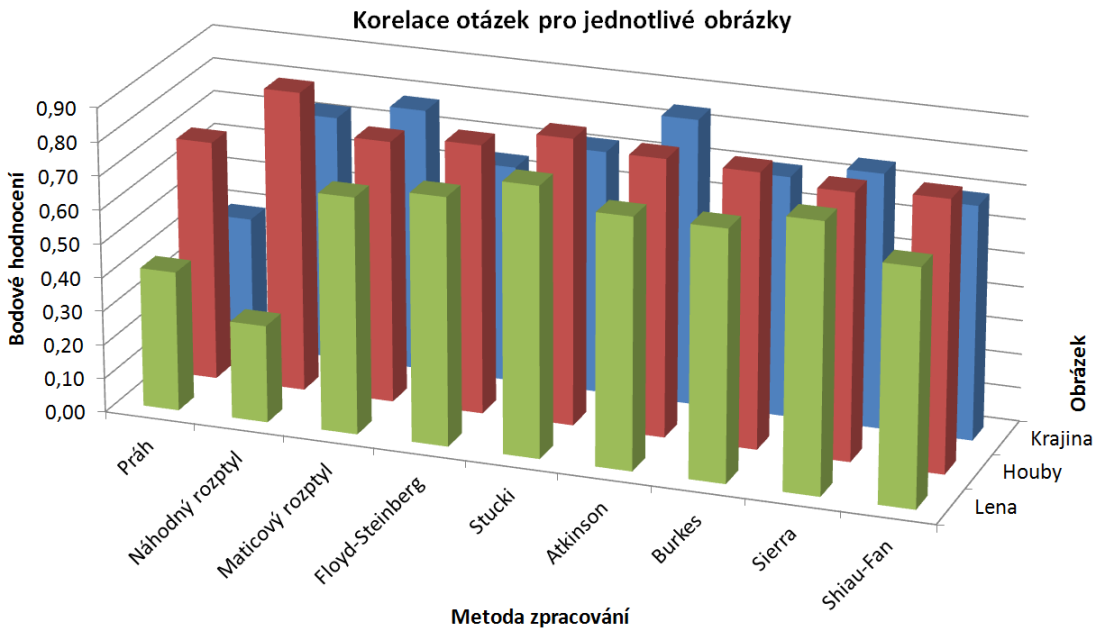


Obr. 4.10: Všechny obrázky – Jak na Vás výsledný obrázek působí?

Tabulka 4.4 a graf 4.10 pro otázku působivosti obrázku ukazuje velmi těsné výsledky tří metod. Nejlépe však podle průměru dopadla metoda Maticového rozptylu, následuje metoda Atkinson a Stucki. Nejhůře na respondenty působila metoda Náhodného rozptylu.

Tab. 4.5: Souhrn korelací obou otázek pro jednotlivé obrázky a metody

Korelace			
Metoda	**Lena**	**Houby**	**Krajina**
Prah	0,41	0,71	0,39
Nahodny	0,29	0,89	0,73
Matice	0,70	0,77	0,78
Floyd	0,73	0,80	0,64
Stucki	0,80	0,85	0,72
Atkinson	0,74	0,82	0,85
Burkes	0,74	0,81	0,71
Sierra	0,79	0,79	0,75
Shiau-Fan	0,70	0,80	0,69

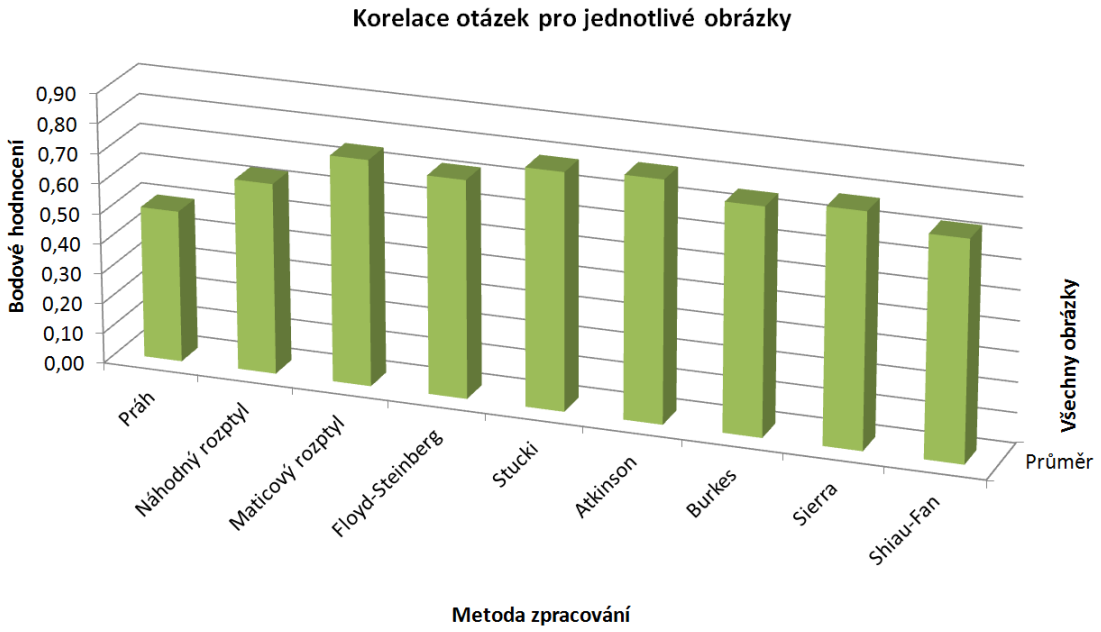


Obr. 4.11: Korelace – všechny obrázky

Korelační tabulka 4.5 a graf 4.11 zobrazují závislosti mezi jednotlivými otázkami. Korelační koeficient nabývá hodnot v rozsahu [-1, 1]. V případě metod distribuce chyby vychází korelační koeficient velmi vysoký. Lze tedy usoudit, že je mezi otázkami přímá závislost. Čím vyšší je podobnost obrázku s originálem, tím více se respondentům obrázek líbí. Nejnižší hodnoty korelační koeficien nabyl u Náhodného rozptylu pro obrázek Lena, z toho lze usoudit mezi shodou a líbivostí nebyla tak vysoká závislost.

Tab. 4.6: Průměr korelací

Korelace	
Metoda	**Všechny obrázky**
Práh	0,50
Náhodný rozptyl	0,63
Maticový rozptyl	0,75
Floyd-Steinberg	0,72
Stucki	0,79
Atkinson	0,80
Burkes	0,75
Sierra	0,78
Shiau-Fan	0,73



Obr. 4.12: Korelace – průměr

V tabulce průměru korelací došlo ke zprůměrovaní korelací pro všechny tři obrázky. Nyní už se u všech metod dostáváme k hodnotám kolem 0,7. To značí vysokou závislost shody a působivosti obrázku. Nižší je pouze u hodnoty pro Prahování. To reflektuje nižší podobnost obrázku s originálem, ale vyšší hodnocení pro působivost obrázku.

Závěr dotazníku tvořily dvě informační otázky. První otázka řešila délku dotazníku. 77% dotazovaných označilo délku dotazníku za normální, 5% za krátký a 19% za příliš dlouhý. Co se týče vhodnosti zvolené délky vyplňování je 8 minut dobře odhadnutá hodnota. V druhé otázce označilo dotazník jako zajímavý 88% respondentů. Respondentů, které dotazník vůbec nezajímal bylo puze 8% a 5% mělo různé připomýnky k vylepšení.

4.2 Subjektivní srovnání – vlastní názor

Jednotlivé metody si dovolím rozdělit do dvou skupin. První skupinu budou tvořit základní metody, tedy prahování, náhodný a maticový rozptyl. Druhou skupinu potom jednotlivé metody distribuce chyby. Uvažovat budu nad černobílými obrázky. Pro více barev je situace jiná a rozdíly mezi jednotlivými distribucemi chyby se stírají. Dovolím si tvrdit, že od 16 barev jsou nerozeznatelné. Srovnávat budu především stejné metody a obrázky, jaké jsou v dotazníku. Dovolím si ale zmínit i obrázky a metody, které v dotazníku chyběly avšak jsou něčím zajímavé.

4.2.1 Jednoduché metody

Prahování

Automatické prahování poskytuje velmi nejisté výsledky. V obraze se ztrácí velmi mnoho informací. Na druhou stranu téměř vždy je z obrázku poznat, co bylo na původním. Manuální práh je samozřejmě závislý podle nastavení. Při nižších hodnotách vynikají tmavé pasáže, ale naprosto se slévají světlé v jednu barvu. Při zvyšování prahové hodnoty je to naopak. Dobrá je ale možnost ovlivnit, co více potřebuji na výsledném obrázku vidět 4.13.

(a) Původní obrázek (b) Práh 0,5

Obr. 4.13: Lena – práh 0,5

Náhodný rozptyl

Náhodný rozptyl je dle mého názoru nejhorší metoda celého srovnání. Snad jen na obrázku Lena je možné odhadnout, co bylo na původním. Metoda na mě nepůsobí

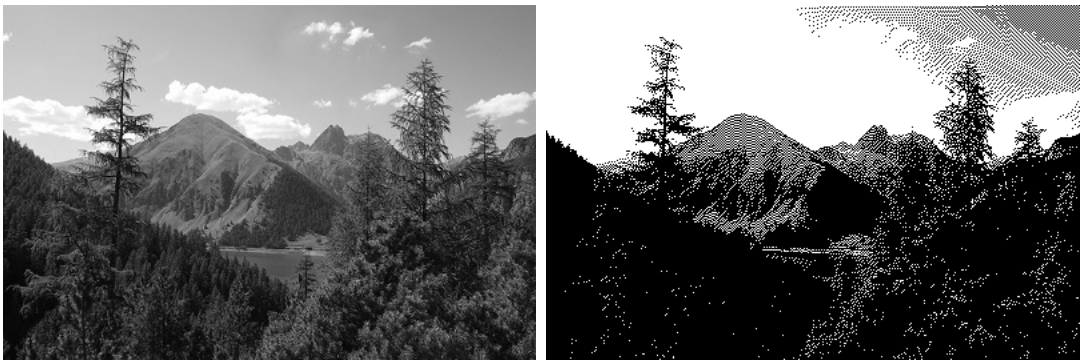
vůbec dobře. Na ukázkovém obrázku není téměř poznat, co bylo na původním 4.14.

(a) Původní obrázek (b) Náhodný rozptyl

Obr. 4.14: Zvíře – náhodný rozptyl

Maticový rozptyl

Maticový rozptyl generuje nejlepší výsledky mezi jednoduchými metodami. Z každého obrázku je jasně vidět, co bylo na původním. Metoda vnáší při dvou barvách do obrazu vysoký kontrast, což hodnotím velmi dobře z estetického hlediska. Bohužel tato metoda se nedokáže vyrovnat s jemnějšími přechody. Vidět je to například na obrázku krajiny, ze kterého se naprosto vytratily mraky z oblohy 4.15.



(a) Původní obrázek (b) Maticový rozptyl

Obr. 4.15: Krajina – maticový rozptyl

4.2.2 Metody distribuce chyby

Floyd-Steinberg

Tato metoda poskytuje velmi dobrý poměr kvality výsledného obrazu k velikosti matice chyb. Matice chyb je nejmenší ze všech zpracovaných metod. Z každého

obrázku je naprosto skvěle poznat, co bylo na původním. Subjektivně ale cítím, že se z obrázku vytrácí kontrast. Nejlépe je ztráta kontrastu vidět na obrázku Lena 4.16.

(a) Původní obrázek (b) Floyd-Steinberg

Obr. 4.16: Lena – Floyd-Steinberg

Jarvis, Stucki, Sierra

Metody Jarvis, Stucki a Sierra si dovolím srovnat dohromady. Ani jedna z metod na mě nepůsobí dobře, všechny vytvářejí v obrazech zvláštní shluky artefaktů. Jednotlivé metody od sebe téměř nelze rozeznat. Všechny totiž mají velmi podobnou matici, jak prvků, tak velikostně. Ani jedna z metod nemá vyšší kontrast obrázku. Samozřejmě ale na všech obrázcích je přesně vidět, co bylo na původním, jak je vidět například u jehličí v obrázku hub 4.17.

(a) Jarvis (b) Stucki (c) Sierra třířádkový

Obr. 4.17: Houby – Jarvis, Stucki, Sierra

Atkinson

Z mého pohledu jasný vítěz mezi všemi metodami. Jednoduchá matice, rychlý výpočet, nejlepší vygenerovaný obraz. Tato metoda vnáší do obrazu zvýšení kontrastu, ale ne nijak přehnané. Nevytváří shluky artefaktů. Vytknout lze snad pouze přechody ve středních tónech. Na všech obrázcích je rozeznatelné vše z originálu. Dle mého názoru tato metoda nejlépe zvládla velmi složitý obrázek zvířat 4.18.

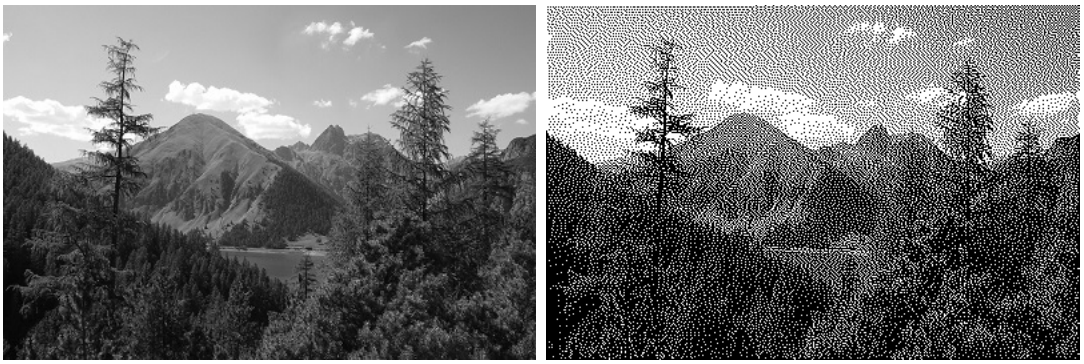
(a) Původní obrázek (b) Atkinson

Obr. 4.18: Zvíře – Atkinson

Burkes

Druhou nejlepší metodou, s poměrně malou maticí a vysokou rychlostí generování obrazu, je metoda Burkes. Přináší mírné zvýšení kontrastu ve výsledném obraze. Osobně bych tuto metodu zařadil na druhé místo především díky shodě obrázku

s originálem. Výstupní obrázky této metody se mi velmi líbí. Kladně hodnotím téměř nulovou přítomnost artefaktů 4.19.

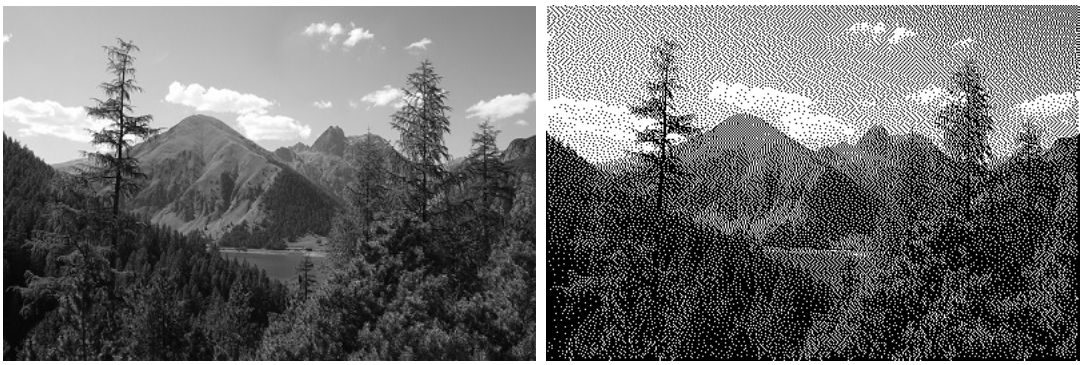


(a) Původní obrázek (b) Burkes

Obr. 4.19: Krajina – Burkes

Shiau-Fan

Poslední metoda generuje překvapivě dobré výsledky. Nezvyšuje kontrast, zachovává hodně detailů. Nicméně generovaný obraz vzhledem k použití variace 3 není nikterak oslnivý. Podobnost obrázku je na vysoké úrovni, navíc metoda do obrazu nezanáší téměř žádné artefakty, což lze hodnotit jedině kladně 4.20.



(a) Původní obrázek (b) Shiau-Fan

Obr. 4.20: Krajina – Shiau-Fan variace 3

5 ZÁVĚR

Diplomová práce objasňuje především metody dithringu obrazu. Pro jejich pochopení bylo nutné v úvodní části zpracovat některé základní pojmy, týkající se digitálního zpracování obrazu. Zásadním pojmem je především barevná paleta.

V druhé části práce jsou jednotlivé metody podrobně rozebrány a srovnány. Jedná se především o teoretické srovnání na základě jejich funkčnosti. Ke každé metodě je pro představu její funkčnosti, ale také kvality zobrazení, uveden výsledný obraz, který metoda generuje. Pro srovnání všech obrazů v plném rozlišení lze nahlédnout do přílohy práce.

Třetí programová část obsahuje návrh java-appletu. Popis funkčnosti, možností a zapracování jednotlivých metod ditheringu obrazu. Popisuje také některá úskalí při vývoji programu, ale také obecné omezení programovacího jazyku Java. Bylo zde dosaženo zapracování veškerých metod, změny směru procházení pixelů a rovnoměrné palety barev. Programu chybí možnost změny počtu barev u dvou metod, které jsou zpracovány a funkční výsledky generují pouze pro dvě barvy.

V poslední části bylo zpracováno subjektivní srovnání metod ditheringu. Subjektivní metodou je myšlen dotazník s výsledky zpracování obrazu jednotlivých metod pomocí naprogramovaného java-appletu. Tyto výsledky byly předložené skupině několika lidí a jeho následně vyhodnoceny. Zahrnuto je i vlastní subjektivní srovnání.

Diplomová práce nabízí možnost pokračování ve vývoji java-appletu. Ten lze více zdokonalit, doprogramovat možnost výběru barev pro všechny metody ditheringu. Jako zásadní vidím opustit programovací jazyk Java a vytvořit webovou aplikaci v HTML5.

LITERATURA

[1] ŽÁRA J., BENEŠ B.,SOCHOR J. FELKEL P. *Moderní počítačová grafika* Druhé vydání. Computer Press, 2005. 608 s. EAN 9788025104545.

[2] LAU L. Daniel, ARCE R. Gonzalo *Modern Digital Halftoning.* Druhé vydání. CRC Press, 2008. 664 s. ISBN-10: 1420047531.

[3] CARRIER, J. *Digital Halftone (Dithering) Methods and Applications.* University of Cambridge, 2008. 19 s.

[4] ŠKVARENINA, Ľ. *Metody ditheringu obrazu: bakalárska práca.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikacií, 2012. 58 s. [cit. 29. 11. 2012].

[5] ŠTĚPÁN, P. *Pokročilé metody ditheringu barevných obrazů, redukce barevného prostoru na n-prvkovou paletu*, diplomová práce, Brno, FIT VUT v Brně, 2011 [cit. 29. 11. 2012].

[6] ŠTEFANÍK, P. *Dithering obrazu: bakalářská práce.* Brno: FEKT VUT v Brně, 2012. 43 stran, [cit. 29. 11. 2012].

[7] *Error diffusion* [online], poslední aktualizace 22.12.2009 [cit. 2.12.2012], Caca Labs. Dostupné z URL: <`http://caca.zoy.org/wiki/libcaca/study/3`>.

[8] *Color quantisation* [online], poslední aktualizace 22.12.2009 [cit. 2.12.2012], CacaLabs. Dostupné z URL: <`http://caca.zoy.org/wiki/libcaca/study/3`>.

[9] BRADLEY, D.; ROTH, G. *Adaptive Thresholding using the Integral Image* [online], 2007.[cit. 2.12.2012] Dostupné z URL: <`http://people.scs.carleton.ca/~roth/iit-publi-cations-iti/docs/gerh-50002.pdf`>.

[10] WOODSIDE, S. *My thresholding is getting pretty good* [online], 2002-2009. [cit. 2.12.2012] Dostupné z URL: <`http://simonwoodside.com/weblog/2004/12/12/my_thresholding_is_getting_pretty_good`>.

[11] *Kvantování*(signál)[online],poslední aktualizace 29.11.2012[cit. 6.12.2012], Wikipedia. Dostupné z URL: <`http://cs.wikipedia.org/wiki-/Kvantování_(signál)`>.

[12] PELIKÁN, J.; SOCHOR, J. *Barva a barevné vidění* [online], poslední aktualizace 30.3.2007 [cit. 6.12.2012]. Dostupné z URL: <`http://www.hluchak.cz/~krhanek/files/Barvy.pdf`>.

[13] ŽÁRA, J. *Zobrazování barev při omezené paletě* [online]. Dostupné z URL: <http://wscg.zcu.cz/WSCG1992/papers92/Zara_92.pdf>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

RGB barevný model – Red(červená), Green(zelená), Blue (modrá)

CMY barevný model – Cyan(azurová), Magenta (purpurová), Yellow(žlutá)

LCD Liguid Crystal Diplay – technologie výroby panelů displejů

DPI Dots per Inch – počet obrazových bodů na palec

IDE Integrated Development Environment – vývojové prostředí

JRE Java Runtime Environment – prostředí pro běh Javy

JDK Java Development Kit – vývojové nástroje

SEZNAM PŘÍLOH

A ORIGINÁLY OBRAZŮ

Obr. A.1: Práh 0,5

Obr. A.2: Náhodný rozptyl

Obr. A.3: Maticový rozptyl

Obr. A.4: Floyd-Steinberg

Obr. A.5: Stucki

Obr. A.6: Atkinson

Obr. A.7: Burkes

Obr. A.8: Sierra třířádkový

Obr. A.9: Shiau-Fan variace 3

B OBSAH CD

- Složka PicturePresenter – projekt Netbeans
- textový soubor s kompletním kódem programu
- PDF soubor výsledné práce

CD obsahuje složku PicturePresenter, což je původní projekt v programu NetBeans. Pro jeho spuštění stačí ve složce dist spustit HTML soubor. Kód je možno prohlédnout v NetBeans 8.0, verze JDK a JRE 8.0 update 05. Grafický návrh proběhl v JavaFX Scene Builder 2.0.

Spuštění HTML je na některých systémech podmíněno úpravou bezpečnosti na hodnotu Medium v souboru javacpl.exe (ProgramFiles>Java>JRE>javacpl.exe)

C ZDROJOVÉ KÓDY

Zdrojový kód metody Floyd-Steinberg

```
static Image floyd(Image input, int ColorsCount)
    {
        return floyd(input, ColorsCount, false, false);
    }
    static Image floyd(Image input, int ColorsCount, boolean
       adaptive_pallete, boolean alternativePassage)
    {
        boolean backStep = false;
        double error = 0.0;
        double temp1 = 0.0;
        double temp2 = 0.0;
        double temp3 = 0.0;
        double temp4 = 0.0;

        WritableImage outputImage = newWritableImage
        ((int)input.getWidth(),(int)input.getHeight());
        PixelWriter pixelWriter = outputImage.getPixelWriter
           ();
        PixelReader originalPixelReader = input.
           getPixelReader();
        WritableImage wInput = new WritableImage((int)input.
           getWidth(),(int)input.getHeight());
        PixelWriter wTempInputPixelWriter = wInput.
           getPixelWriter();

        for (int copyY = 0; copyY < wInput.getHeight(); copyY
           ++) //kopie obrázku
        {
            for (int copyX = 0; copyX < wInput.getWidth();
               copyX++)
            {
                wTempInputPixelWriter.setColor(copyX,copyY,
                   originalPixelReader.getColor( copyX, copyY
                   ));
            }
        }
        PixelReader pixelReader = wInput.getPixelReader();
```

```
//cyklus průchodu obrazem
for (int readY = 0; readY < input.getHeight(); readY
   ++)
{
    for (int X = 0; X < input.getWidth(); X++)
    {
        int readX=0; //směr průchodu obrazem
        if(backStep)
        {
         readX=(int)(outputImage.getWidth()-1-X);
        }
        else
        {
         readX = X;
        }

        Color color = pixelReader.getColor(readX,
           readY);
        double actual_pixel=color2Double(color);

        double[] reduce_with_error = reduceColor(
           actual_pixel, ColorsCount,
           adaptive_pallete);
        pixelWriter.setColor(readX,readY,
        double2Color(reduce_with_error[0]));
        error = reduce_with_error[1];
        //průchod po pixelech
        if( readX < input.getWidth()-1 )
        {
          //určení pozic pixelu
          temp1 = color2Double(pixelReader.getColor(
             readX+1, readY));
          //výpočet chyby
          temp1 += error *(7./16.);
          //zapsání chyby na pozici
          wTempInputPixelWriter.setColor(readX+1,
             readY, double2Color(temp1));
        }
```

```
                if( readY < input.getHeight() -1 )
                {
                    temp2 = color2Double(pixelReader.getColor
                       (readX, readY+1));
                    temp2 += error* (5./16.);
                    wTempInputPixelWriter.setColor(readX,
                       readY+1, double2Color(temp2));
                    if( readX < input.getWidth() -1 )
                    {
                       temp3 = color2Double(pixelReader.
                          getColor(readX+1, readY+1));
                       temp3 += error*(1./16.);
                       wTempInputPixelWriter.setColor(readX
                          +1,
                       readY+1, double2Color(temp3));
                    }
                    if( readX > 0 )
                    {
                       temp4 = color2Double(pixelReader.
                          getColor(readX-1, readY+1));
                       temp4 += error *(3./16.);
                       wTempInputPixelWriter.setColor(readX
                          -1,readY+1,double2Color(temp4));
                    }
                }
            }
            if(alternativePassage)
            {
                backStep = ! backStep;
            }
        }
        return outputImage;
    }
```